Chapter 23: UPS Command Reference

This chapter contains full usage information on all the **UPS** commands. In particular, for each command you will find:

- a statement of the purpose and/or function of the command
- the command syntax
- a listing of commonly used options, without descriptions
- a listing of all valid options, with command-specific descriptions
- (as needed) a section called “Options Valid with -G”
- (as needed) a section called “More Detailed Description” which typically includes:
  - detailed command-specific usage information
  - information on environment variables that affect execution of the command or are affected by it
  - a list of internal functions the command performs (if more extensive than suggested by the command description)
- command examples

23.1 setup

Issue the **`setup`** command for a product prior to invoking the product. The **`setup`** command performs the necessary operations in your login environment to make an installed, declared product instance accessible to you. Typically, the operations include modifying environment variables or adding to your $PATH.

23.1.1 Command Syntax

```
% setup [<options>] <product> [<version>]
```

23.1.2 Commonly Used Options

See section 23.1.3 *All Valid Options* for descriptions of each option.

`-f <flavor>` Or one of **`-0`**, **`-1`**, **`-2`**, **`-3`**, **`-4`**, or **`-H`** (alone or together with one of **`-0`**, **`-1`**, **`-2`**, **`-3`**, **`-4`**)

`-g <chainName>` Or one of **`-c`**, **`-d`**, **`-n`**, **`-o`**, **`-t`**

`-q <qualifierList>`

`-z <databaseList>`

Table 23.1.2-a:

`-f <flavor>`	Or one of **`-0`**, **`-1`**, **`-2`**, **`-3`**, **`-4`**, or **`-H`** (alone or together with one of **`-0`**, **`-1`**, **`-2`**, **`-3`**, **`-4`**)
`-g <chainName>`	Or one of **`-c`**, **`-d`**, **`-n`**, **`-o`**, **`-t`**
`-q <qualifierList>`	
`-z <databaseList>`	

23.1.3 All Valid Options

`-?` (**`"-?"`** for **csh**) Prints command description and option usage information to screen

`-B <depProdName>= "<options>"`

Specifies options to prepend to the `setupRequired` line (in table file) for the dependent product **`<depProdName>`**

`-c` Finds product instance chained to “current”

`-d` Finds product instance chained to “development”

`-e` Sets $UPS_EXTENDED (to the value **`1`**). This is useful for SETUP actions which call scripts.

`-f <flavor>` Described below under “The flavor options”.

`-g <chainName>` Finds product instance chained to **`<chainName>`**

`-H <flavor>` Described below under “The flavor options”.

`-j` Ignores dependencies, sets up just specified top-level product

-k Prevents execution of unsetup files prior to (subsequent) setup

-m <tableFileName> Specifies table file name

-M <tableFileDir> Specifies table file directory

-n Finds product instance chained to “new”

-o Finds product instance chained to “old”

-O "<flags>" Sets the value of $UPS_OPTIONS to **<flags>**. This is useful for SETUP actions which call scripts.

-P Requires **UPS** to rely only on information supplied on the command line to locate the product instance (prevents **UPS** from searching in a database)

-q <qualifierList> Finds product instance with the specified qualifiers (required and/or optional)

-r <prodRootDir> Specifies the product root directory

-R Sets up only the required (non-optional) dependencies.

-s Lists what command would do; but does not execute the command

-t Specifies product instance chained to “test”

-U <upsDir> Specifies location of `ups` directory; default value is **ups** (relative to the product root directory)

-v(**vvv**) Prints out extra debugging information.

-V Does not delete the temporary script files or partially installed products when command finishes; instead lists them on the screen

-z <databaseList> Specifies the database(s) in which to look for the product and its dependencies

-Z Times the command (does not include time for sourcing of temp file for **setup**)

Table 23.1.3-a:

-? ("-?" for csh)	Prints command description and option usage information to screen
-B <dep-Prod-Name>= "<options>"	Specifies options to prepend to the `setupRequired` line (in table file) for the dependent product **<depProdName>**

Table 23.1.3-a:

-c	Finds product instance chained to “current”
-d	Finds product instance chained to “development”
-e	Sets $UPS_EXTENDED (to the value **1**). This is useful for SETUP actions which call scripts.
-f <flavor>	Described below under “The flavor options”.
-g <chainName>	Finds product instance chained to **<chainName>**
-H <flavor>	Described below under “The flavor options”.
-j	Ignores dependencies, sets up just specified top-level product
-k	Prevents execution of unsetup files prior to (subsequent) setup
-m <tableFileName>	Specifies table file name
-M <tableFileDir>	Specifies table file directory
-n	Finds product instance chained to “new”
-o	Finds product instance chained to “old”
-O "<flags>"	Sets the value of $UPS_OPTIONS to **<flags>**. This is useful for SETUP actions which call scripts.
-P	Requires **UPS** to rely only on information supplied on the command line to locate the product instance (prevents **UPS** from searching in a database)
-q <qualifierList>	Finds product instance with the specified qualifiers (required and/or optional)
-r <prodRootDir>	Specifies the product root directory
-R	Sets up only the required (non-optional) dependencies.
-s	Lists what command would do; but does not execute the command
-t	Specifies product instance chained to “test”
-U <upsDir>	Specifies location of **ups** directory; default value is **ups** (relative to the product root directory)
-v(vvv)	Prints out extra debugging information.

Table 23.1.3-a:

-V	Does not delete the temporary script files or partially installed products when command finishes; instead lists them on the screen
-z <dataBaseList>	Specifies the database(s) in which to look for the product and its dependencies
-Z	Times the command (does not include time for sourcing of temp file for **setup**)

The flavor options

Flavor may be specified using **-f**, using **-H** by itself or in combination with any of **-0**, **-1**, **-2**, **-3**, **-4**, or just using one of **-0**, **-1**, **-2**, **-3**, **-4**. These options are not valid with each other (except **-H** with a number option).

If a dependency is specified in the table file with a particular flavor, the flavor specified on the command line is ignored for that dependency.

-f <flavor> Finds product instance of specified flavor. If specified and no exact match is found, the command fails. Multiple values accepted, but **UPS** looks only at first in list.

-H <flavor> Specifies flavor and builds a flavor list for that family starting at the level specified. Multiple values accepted, but **UPS** looks only at first in list.

Can be used alone (without an accompanying number option). In this case, **UPS** finds the best match instance for the specified flavor family.

If used with any of **-0**, **-1**, **-2**, **-3**, **-4**, **UPS** finds the product instance of specified level of that flavor; e.g., **-2H IRIX+6.2** is equivalent to **-f IRIX+6**.

-0 Specifies flavor as NULL; equivalent to **-f NULL**

-1 Specifies flavor as OS value up to the generic OS; e.g., on a SunOS machine it is equivalent to **-f SunOS**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX.

-2 Specifies flavor for product instance on local and distribution nodes as OS value up to the version; e.g., equivalent to **-f SunOS+5**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.

-3 Specifies flavor for product instance on local and distribution nodes upto the release of the version; e.g., equivalent to **-f SunOS+5.6**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.2.

-4	Specifies flavor for product instance on local and distribution nodes up to the patch number of the release; e.g., equivalent to **-f SunOS+5.6.2**; if given together with **-H IRIX+6.2.1**, flavor is then specified as IRIX+6.2.1.

Table 23.1.3-b:

-f <flavor>	Finds product instance of specified flavor. If specified and no exact match is found, the command fails. Multiple values accepted, but **UPS** looks only at first in list.
-H <flavor>	Specifies flavor and builds a flavor list for that family starting at the level specified. Multiple values accepted, but **UPS** looks only at first in list. Can be used alone (without an accompanying number option). In this case, **UPS** finds the best match instance for the specified flavor family. If used with any of **-0**, **-1**, **-2**, **-3**, **-4**, **UPS** finds the product instance of specified level of that flavor; e.g., **-2H IRIX+6.2** is equivalent to **-f IRIX+6**.
-0	Specifies flavor as NULL; equivalent to **-f NULL**
-1	Specifies flavor as OS value up to the generic OS; e.g., on a SunOS machine it is equivalent to **-f SunOS**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX.
-2	Specifies flavor for product instance on local and distribution nodes as OS value up to the version; e.g., equivalent to **-f SunOS+5**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.
-3	Specifies flavor for product instance on local and distribution nodes upto the release of the version; e.g., equivalent to **-f SunOS+5.6**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.2.
-4	Specifies flavor for product instance on local and distribution nodes upto the patch number of the release; e.g., equivalent to **-f SunOS+5.6.2**; if given together with **-H IRIX+6.2.1**, flavor is then specified as IRIX+6.2.1.

23.1.4 More Detailed Description

In general, **UPS** products require that the **setup** command be issued on a product instance before invoking it (unless it is a dependent product of one that is already setup). The setup processes are intended to make the appropriate changes to your software environment in order to make the requested product available for use.

Only one instance of a product can be setup at a time. Each time you run **setup** on an additional instance of the same product, the previously active instance is automatically unsetup first.

Internal Processes

- Check node authorization
- If necessary, process UNSETUP action
- Process SETUP action
- Source the temp file

Environment Variables Set by Default During setup

When an instance is setup, either or both of the two environment variables $<PRODUCT>_DIR and $SETUP_<PRODUCT> may get defined. By default, both do.

$<PRODUCT>_DIR	points to the root directory of the product instance selected by the **`setup`** command[1]
$SETUP_<PRODUCT>	a string containing all the information that the **`unsetup`** command needs in order to identify the instance when it is time to remove access to the product

1. In versions of **UPS** previous to v4, at **`unsetup`** time **UPS** matched the current definition of $<PRODUCT>_DIR with the product's versions in the database. If no match was found, it assumed `$<PRODUCT>_DIR/ups` as the location of the unsetup script.
In **UPS** v4, $<PRODUCT>_DIR may not be set because it is no longer a requirement. The unsetup action is generally not performed via a script, but rather via functions in a table file, and this table file is not constrained to reside under the product root directory.

In both of them, <PRODUCT> is the name of the product in upper case. We'll use the product **cern** as an example to show you typical values for these variables:

```
% setup cern
% echo $CERN_DIR
   /afs/fnal.gov/products/SunOS5/cern/v97aa
% echo $SETUP_CERN
   cern          v97aa          -f          SunOS+5          -z
   /usr/upsII/ups_database/declared/afs
```

Use of the $SETUP_<PRODUCT> Variable by unsetup

`unsetup` uses the environment variable $SETUP_<PRODUCT>, by default, to determine which instance to unsetup. If this variable was *not* set during product setup (i.e., the **`setup`** default functions were not run, or **`setupEnv()`** was not run), then when you run **`unsetup`**, you must specify on the command line which instance to unsetup; running simply **`unsetup <product>`** causes no action to be taken. See *Use of the $SETUP_<PRODUCT> Variable* under section 23.2 *unsetup* for more information regarding the **`unsetup`** command.

23.1.5 setup Examples

In the following examples, when we say that all dependencies of a product get set up, we mean all except optional dependencies that are unavailable.

Setup default instance of product

```
% setup xemacs
```

This sets up the current instance of the product **xemacs** for the best match flavor of your OS. If the product has any dependencies, they get setup too, by default.

```
% setup -v xemacs
```

This command sets up the same instance as above, but displays verbose information (usually used for debugging, but useful to see what's going on). If any file that is "opened for read" does not exist, you'll see `ERROR` at the end of the line. This is often but not always a fatal error. The output looks like this:

```
   UPSFIL:
   /usr/upsII/ups_database/declared/oss/.upsfiles/dbconfig    -
   Open file for read
```

```
UPSFIL:
/usr/upsII/ups_database/declared/oss/xemacs/current.chain  -
Open file for read
UPSFIL:
/usr/upsII/ups_database/declared/oss/xemacs/current.chain  -
Read 2 item(s)
UPSFIL:
/usr/upsII/ups_database/declared/oss/xemacs/v19_14.version -
Open file for read
UPSFIL:
/usr/upsII/ups_database/declared/oss/xemacs/v19_14.version -
Read 2 item(s)
UPSFIL:
/usr/upsII/ups_database/declared/oss/xemacs/v19_14.table   -
Open file for read
UPSFIL:
/usr/upsII/ups_database/declared/oss/xemacs/v19_14.table   -
Read 4 item(s)
UPSFIL:
/usr/upsII/ups_database/devel/new/xemacs/current.chain     -
Open file for read ERROR
```

Restrict the setup of dependent products

```
% setup -R exmh
```

Use of the **-R** option sets up the specified product and its required dependencies only.

```
% setup -j exmh
```

Use of the **-j** option sets up only the specified product; none of its dependencies get setup.

Setup a chained instance (other than the default “current”)

```
% setup -t tex
```

```
% setup -g test tex
```

Either of these commands sets up the instance of **tex** chained to “test” (for the default flavor). To setup any chained instance other than current, include the chain flag in the command.

Setup a product specifying its version

```
% setup tex v3_1415a
```

This command sets up version v3_1415a of **tex** whether or not it has a chain. Run a **ups list** command to get the version information.

Setup a product declared with qualifiers

```
% setup -q BUILD prod1
```

This command sets up the current instance of **prod1** for the operating system on which you're working, along with its build dependencies (assuming the qualifier BUILD has been implemented in the product files in the standard way, see section 17.2.3 *Products Requiring Build (In-House and Third-Party)*).

Remember that qualifiers are case-sensitive.

Setup a product and activate extended functionality

To setup the instance of product **prod1** chained to development, and all of **prod1**'s dependencies, and to activate extended setup actions, enter:

```
% setup -d -e prod1
```

The **`-e`** option sets $UPS_EXTENDED on for **prod1** and for any of its **UPS** product requirements that were declared with the **`-e`** option. This is used to activate any extended functionality the product provider may have included in the setup action for this instance (e.g., defining extra environment variables).

Setup a product directly from CD-ROM

If you have a CD-ROM image that includes products that are unwound and ready-to-use, you can run them directly off the CD-ROM without downloading them to your system.

First insert the CD-ROM and run the **`mount`** command:

```
% mount -t iso9660 /dev/cdrom /path/to/db/on/cdrom
```

Then run the **`setup`** command. **UPS** needs to look in the right database, so either add it to your $PRODUCTS variable beforehand, or supply it on the command line, e.g.,:

```
% setup -z /path/to/db/on/cdrom <product> <version>
```

More information on **UPS** product distribution CD-ROMs can be found at `http://www.fnal.gov/docs/products/ups/ReferenceManual/misc/cdrom.html`.

23.2 unsetup

The **`unsetup`** command makes the specified product no longer available for use. It undoes the changes made to the environment by **`setup`**. You may need to explicitly unsetup a **UPS** product if you are short on environment

variable space and want to get rid of extra environment variables or shorten the $PATH variable length; otherwise you typically don't need to run this command.

23.2.1 Command Syntax

```
% unsetup [<options>] <product> [<version>]
```

23.2.2 All Valid Options

Typically, this command is issued with no options.

-? (**"-?"** for **csh**)	Prints command description and option usage information to screen
-c	Finds product instance chained to "current"
-d	Finds product instance chained to "development"
-e	Sets $UPS_EXTENDED (to the value **1**).
-f <flavor>	Described below under "The flavor options".
-g <chainName>	Finds product instance chained to **<chainName>**
-H <flavor>	Described below under "The flavor options".
-j	Ignores dependencies, runs **unsetup** on just on top level product
-m <tableFileName>	Specifies table file name
-M <tableFileDir>	Specifies table file directory
-n	Finds product instance chained to "new"
-o	Finds product instance chained to "old"
-O "<flags>"	Sets the value of $UPS_OPTIONS to **<flags>**. (This option is ignored.)
-P	Requires **UPS** to rely only on information supplied on the command line to locate the product instance (prevents **UPS** from searching in a database)
-q <qualifierList>	Finds product instance with the specified qualifiers (required and/or optional)
-r <prodRootDir>	Specifies the product root directory (This option is ignored.)
-s	Lists what command would do; but does not execute the command
-t	Finds product instance chained to "test"

-U <upsDir> Specifies location of `ups` directory; default value is **ups**

-v(**vvv**) Prints out extra debugging information.

-V Does not delete the temporary script files or partially installed products when command finishes; instead lists them on the screen

-z <databaseList> Specifies the database(s) in which the product(s) are declared

-Z Times the command (does not include time for sourcing of temp file for **setup/unsetup**)

Table 23.2.2-a:

-? ("-?" for csh)	Prints command description and option usage information to screen
-c	Finds product instance chained to “current”
-d	Finds product instance chained to “development”
-e	Sets $UPS_EXTENDED (to the value **1**).
-f <flavor>	Described below under “The flavor options”.
-g <chainName>	Finds product instance chained to **<chainName>**
-H <flavor>	Described below under “The flavor options”.
-j	Ignores dependencies, runs **unsetup** on just on top level product
-m <tableFileName>	Specifies table file name
-M <tableFileDir>	Specifies table file directory
-n	Finds product instance chained to “new”
-o	Finds product instance chained to “old”
-O "<flags>"	Sets the value of $UPS_OPTIONS to **<flags>**. (This option is ignored.)
-P	Requires **UPS** to rely only on information supplied on the command line to locate the product instance (prevents **UPS** from searching in a database)

Table 23.2.2-a:

-q <qualifier-List>	Finds product instance with the specified qualifiers (required and/or optional)
-r <prodRootDir>	Specifies the product root directory (This option is ignored.)
-s	Lists what command would do; but does not execute the command
-t	Finds product instance chained to "test"
-U <upsDir>	Specifies location of `ups` directory; default value is **ups**
-v(vvv)	Prints out extra debugging information.
-V	Does not delete the temporary script files or partially installed products when command finishes; instead lists them on the screen
-z <databaseList>	Specifies the database(s) in which the product(s) are declared
-Z	Times the command (does not include time for sourcing of temp file for **setup/unsetup**)

The flavor options

Flavor may be specified using **-f**, using **-H** by itself or in combination with any of **-0**, **-1**, **-2**, **-3**, **-4**, or just using one of **-0**, **-1**, **-2**, **-3**, **-4**. These options are not valid with each other (except **-H** with a number option).

If a dependency is specified in the table file with a particular flavor, the flavor specified on the command line is ignored for that dependency.

-f <flavor> Finds product instance of specified flavor. If specified and no exact match is found, the command fails. Multiple values accepted, but **UPS** looks only at first in list.

-H <flavor> Specifies flavor and builds a flavor list for that family starting at the level specified. Multiple values accepted, but **UPS** looks only at first in list.

Can be used alone (without an accompanying number option). In this case, **UPS** finds the best match instance for the specified flavor family.

If used with any of **-0**, **-1**, **-2**, **-3**, **-4**, **UPS** finds the product instance of specified level of that flavor; e.g., **-2H IRIX+6.2** is equivalent to **-f IRIX+6**.

-0 Specifies flavor as NULL; equivalent to **-f NULL**

-1 Specifies flavor as OS value up to the generic OS; e.g., on a SunOS machine it is equivalent to **-f SunOS**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX.

-2 Specifies flavor for product instance on local and distribution nodes as OS value up to the version; e.g., equivalent to **-f SunOS+5**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.

-3 Specifies flavor for product instance on local and distribution nodes upto the release of the version; e.g., equivalent to **-f SunOS+5.6**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.2.

-4 Specifies flavor for product instance on local and distribution nodes upto the patch number of the release; e.g., equivalent to **-f SunOS+5.6.2**; if given together with **-H IRIX+6.2.1**, flavor is then specified as IRIX+6.2.1.

Table 23.2.2-b:

-f <flavor>	Finds product instance of specified flavor. If specified and no exact match is found, the command fails. Multiple values accepted, but **UPS** looks only at first in list.
-H <flavor>	Specifies flavor and builds a flavor list for that family starting at the level specified. Multiple values accepted, but **UPS** looks only at first in list. Can be used alone (without an accompanying number option). In this case, **UPS** finds the best match instance for the specified flavor family. If used with any of **-0**, **-1**, **-2**, **-3**, **-4**, **UPS** finds the product instance of specified level of that flavor; e.g., **-2H IRIX+6.2** is equivalent to **-f IRIX+6**.
-0	Specifies flavor as NULL; equivalent to **-f NULL**
-1	Specifies flavor as OS value up to the generic OS; e.g., on a SunOS machine it is equivalent to **-f SunOS**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX.
-2	Specifies flavor for product instance on local and distribution nodes as OS value up to the version; e.g., equivalent to **-f SunOS+5**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.
-3	Specifies flavor for product instance on local and distribution nodes upto the release of the version; e.g., equivalent to **-f SunOS+5.6**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.2.
-4	Specifies flavor for product instance on local and distribution nodes up to the patch number of the release; e.g., equivalent to **-f SunOS+5.6.2**; if given together with **-H IRIX+6.2.1**, flavor is then specified as IRIX+6.2.1.

23.2.3 More Detailed Description

`unsetup` is intended to undo the changes to your software environment made during product setup. It makes the product no longer available for use. You may need to explicitly unsetup a **UPS** product if you are short on environment variable space and want to get rid of extra environment variables or shorten the $PATH variable length. Unsetup gets done automatically for you when you setup a different instance of the same product.

When you no longer need to access a product, in most cases you can simply type:

```
% unsetup <product>
```

for example:

```
% unsetup tex
```

Sometimes this isn't sufficient. The $SETUP_<PRODUCT> variable governs the behavior, as described below.

Use of the $SETUP_<PRODUCT> Variable

`unsetup` uses the environment variable $SETUP_<PRODUCT>, by default, to determine which instance to unsetup. If this variable was *not* set during product setup (i.e., the **`setup`** default functions were not run, or **`setupEnv()`** was not run), then you must specify on the command line which instance to unsetup; running simply **`unsetup <product>`** causes no action to be taken.

If $SETUP_<PRODUCT> has been set (the usual case), it is best to run **`unsetup`** with no options (except possibly **`-j`** as discussed below). If any instance-identifying information besides product name is specified on the **`unsetup`** command line, this information gets ignored.

Behavior of unsetup for Product Dependencies

The behavior of **`unsetup`** as regards product dependencies depends upon a couple of factors:

- whether an UNSETUP action exists in the main product's table file
- whether $SETUP_<PRODUCT> has been defined for the product dependency

If ACTION=UNSETUP is defined for the main product, then[1]:

1) if it includes the function **`unsetupRequired`** for the dependency

1. In the following numbered list, the same is true for the function **`unsetupOptional`**, with the difference that no failure will occur if the specified instance is not available to be unsetup.

with *no* instance-identifying information (e.g., **`unsetupRequired (<dep_product>)`** only; no options or version), and if $SETUP_<PRODUCT> is defined for the dependency, **`unsetup`** will be run on the instance identified by $SETUP_<PRODUCT>.

2) if it includes the function **`unsetupRequired`** for the dependency with *no* instance-identifying information (e.g., **`unsetupRequired (<dep_product>)`** only; no options or version), and if $SETUP_<PRODUCT> is *not* defined for the dependency, no **`unsetup`** will be performed for the dependency.

3) if it includes the function **`unsetupRequired`** for the dependency with some instance-identifying information (e.g., **`unsetupRequired -q "build" <dep_product>`**), then **`unsetup`** is run on this specified instance of the product dependency; if $SETUP_<PRODUCT> is defined for the dependency, it is ignored.

If ACTION=UNSETUP is *not* defined for the main product, then:

1) if $SETUP_<PRODUCT> is defined for the product dependency, then **`unsetup`** will be run on the instance identified by $SETUP_<PRODUCT>.

2) if $SETUP_<PRODUCT> is *not* defined for the dependency, no **`unsetup`** will be performed for the dependency.

If you use the **`-j`** option in the **`unsetup`** command of the main product, only the main product gets unsetup; its product dependencies are left untouched.

Internal Processes

- Check node authorization
- Process UNSETUP action
- Source the temp file

The UNSETUP default functions are to undo the default SETUP functions (i.e., unset $<PRODUCT>_DIR and $SETUP_<PRODUCT>).

Note: If there is no UNSETUP action, then **`unsetup`** undoes everything done in SETUP action. However, if SETUP includes non-reversible functions, these cannot be undone by **`unsetup`**.

23.2.4 unsetup Examples

```
% unsetup tex
```

This command unsets the product **tex**. When you no longer need to access a product, in most cases you can simply use the product name to identify it.

```
% unsetup -j netscape
```

The **`-j`** option in this command causes **UPS** to unsetup the product **netscape**, while leaving all its dependencies setup.

23.3 ups configure

For any product instance whose table file includes a CONFIGURE action, the **ups configure** command executes this action. A CONFIGURE action usually includes functions to construct symbolic links, copy files, or perform automatic local customization of the product. The **ups configure** command gets run by default by **ups declare** when the product is initially declared to a database (see section 23.5 *ups declare*, in particular the **-C** option), but can be run manually as needed (e.g., on nodes of different flavors).

23.3.1 Command Syntax

```
% ups configure [<options>] <product> [<version>]
```

23.3.2 Commonly Used Options

-f <flavor> Or one of **-0**, **-1**, **-2**, **-3**, **-4**, or **-H** (alone or together with one of **-0**, **-1**, **-2**, **-3**, **-4**)

-g <chainName> Or one of **-c**, **-d**, **-n**, **-o**, **-t**

-q <qualifierList>

-z <databaseList>

Table 23.3.2-a:

-f <flavor>	Or one of **-0**, **-1**, **-2**, **-3**, **-4**, or **-H** (alone or together with one of **-0**, **-1**, **-2**, **-3**, **-4**)
-g <chainName>	Or one of **-c**, **-d**, **-n**, **-o**, **-t**
-q <qualifierList>	
-z <databaseList>	

23.3.3 All Valid Options

-? (**"-?"** for **csh**) Prints command description and option usage information to screen

-c Finds product instance chained to “current”

-d Finds product instance chained to “development”

-f <flavor> Described below under “The flavor options”.

-g <chainName> Finds product instance chained to **<chainName>**

-H <flavor> Described below under “The flavor options”.

-m <tableFileName> Specifies table file name

-M <tableFileDir> Specifies table file directory

-n Finds product instance chained to “new”

-o Finds product instance chained to “old”

-O "<flags>" Sets the value of $UPS_OPTIONS to **<flags>**.

-P Requires **UPS** to rely only on information supplied on the command line to locate the product instance (prevents **UPS** from searching in a database)

-q <qualifierList> Finds product instance with the specified qualifiers (required and/or optional)

-r <prodRootDir> Specifies the product root directory

-s Lists what command would do; but does not execute the command

-t Finds product instance chained to “test”

-U <upsDir> Specifies location of `ups` directory; default value is **ups**

-v(**vvv**) Prints out extra debugging information.

-V Does not delete the temporary script files or partially installed products when command finishes; instead lists them on the screen

-z <databaseList> Specifies the database(s) in which to look for the product and its dependencies

-Z Times the command (does not include time for sourcing of temp file for **setup/unsetup**)

Table 23.3.3-a:

-? ("-?" for csh)	Prints command description and option usage information to screen
-c	Finds product instance chained to “current”
-d	Finds product instance chained to “development”

Table 23.3.3-a:

-f <flavor>	Described below under “The flavor options”.
-g <chainName>	Finds product instance chained to **<chainName>**
-H <flavor>	Described below under “The flavor options”.
-m <tableFileName>	Specifies table file name
-M <tableFileDir>	Specifies table file directory
-n	Finds product instance chained to “new”
-o	Finds product instance chained to “old”
-O "<flags>"	Sets the value of $UPS_OPTIONS to **<flags>**.
-P	Requires **UPS** to rely only on information supplied on the command line to locate the product instance (prevents **UPS** from searching in a database)
-q <qualifierList>	Finds product instance with the specified qualifiers (required and/or optional)
-r <prodRootDir>	Specifies the product root directory
-s	Lists what command would do; but does not execute the command
-t	Finds product instance chained to “test”
-U <upsDir>	Specifies location of **ups** directory; default value is **ups**
-v(vvv)	Prints out extra debugging information.
-V	Does not delete the temporary script files or partially installed products when command finishes; instead lists them on the screen
-z <databaseList>	Specifies the database(s) in which to look for the product and its dependencies
-Z	Times the command (does not include time for sourcing of temp file for **setup/unsetup**)

The flavor options

Flavor may be specified using **-f**, using **-H** by itself or in combination with any of **-0**, **-1**, **-2**, **-3**, **-4**, or just using one of **-0**, **-1**, **-2**, **-3**, **-4**. These options are not valid with each other (except **-H** with a number option).

-f <flavor> Finds product instance of specified flavor. If specified and no exact match is found, the command fails. Multiple values accepted, but **UPS** looks only at first in list.

-H <flavor> Specifies flavor and builds a flavor list for that family starting at the level specified. Multiple values accepted, but **UPS** looks only at first in list.

Can be used alone (without an accompanying number option). In this case, **UPS** finds the best match instance for the specified flavor family.

If used with any of **-0**, **-1**, **-2**, **-3**, **-4**, **UPS** finds the product instance of specified level of that flavor; e.g., **-2H IRIX+6.2** is equivalent to **-f IRIX+6**.

-0 Specifies flavor as NULL; equivalent to **-f NULL**

-1 Specifies flavor as OS value up to the generic OS; e.g., on a SunOS machine it is equivalent to **-f SunOS**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX.

-2 Specifies flavor for product instance on local and distribution nodes as OS value up to the version; e.g., equivalent to **-f SunOS+5**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.

-3 Specifies flavor for product instance on local and distribution nodes upto the release of the version; e.g., equivalent to **-f SunOS+5.6**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.2.

-4 Specifies flavor for product instance on local and distribution nodes upto the patch number of the release; e.g., equivalent to **-f SunOS+5.6.2**; if given together with **-H IRIX+6.2.1**, flavor is then specified as IRIX+6.2.1.

Table 23.3.3-b:

-f <flavor>	Finds product instance of specified flavor. If specified and no exact match is found, the command fails. Multiple values accepted, but **UPS** looks only at first in list.

Table 23.3.3-b:

-H <flavor>	Specifies flavor and builds a flavor list for that family starting at the level specified. Multiple values accepted, but **UPS** looks only at first in list. Can be used alone (without an accompanying number option). In this case, **UPS** finds the best match instance for the specified flavor family. If used with any of **-0**, **-1**, **-2**, **-3**, **-4**, **UPS** finds the product instance of specified level of that flavor; e.g., **-2H IRIX+6.2** is equivalent to **-f IRIX+6**.
-0	Specifies flavor as NULL; equivalent to **-f NULL**
-1	Specifies flavor as OS value up to the generic OS; e.g., on a SunOS machine it is equivalent to **-f SunOS**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX.
-2	Specifies flavor for product instance on local and distribution nodes as OS value up to the version; e.g., equivalent to **-f SunOS+5**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.
-3	Specifies flavor for product instance on local and distribution nodes upto the release of the version; e.g., equivalent to **-f SunOS+5.6**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.2.
-4	Specifies flavor for product instance on local and distribution nodes upto the patch number of the release; e.g., equivalent to **-f SunOS+5.6.2**; if given together with **-H IRIX+6.2.1**, flavor is then specified as IRIX+6.2.1.

23.3.4 More Detailed Description

Installation/configuration procedures that can be completely automated are typically collected in the table file in a CONFIGURE action (*actions* are described in Chapter 34: *Actions and ACTION Keyword Values*), or in a script called `configure` which is called from the table file. The configuration may involve creating links to the product root directory from other areas. If the area is not identical for each machine flavor accessing the **UPS** database in which the product instance has been declared (i.e., the same path but separate areas), then you will need to run the **ups configure** command manually once per flavor, on a node of that flavor. If each node mounts a unique area, you generally have to run special commands (e.g., **ups install**, **ups initialize**, etc.) that are documented in the product's `INSTALL_NOTE` file. If you are not sure whether you need to configure a product instance on each flavor/node, look through the configuration steps in the table file to see what they do.

Internal Processes

- Check node authorization
- Process CONFIGURE action
- Execute temp file

23.3.5 ups configure Examples

perl is a product that requires **`ups configure`** to be run manually for each machine flavor in a cluster. The sample command, which should be issued from a machine of flavor SunOS+5, runs the CONFIGURE action in the table file associated with the product **perl**, version v5_005 for flavor SunOS+5.

```
% ups configure perl v5_005 -f SunOS+5
```

This command should take care of the configuration of **perl** on all the machines of flavor SunOS+5 in the cluster. A command like this, but with the appropriate flavor, must be run for each machine flavor represented in the cluster.

23.4 ups copy

The **`ups copy`** command was designed as a **UPS** product development tool allowing a new instance of a product to be declared "like" another.

23.4.1 Command Syntax

```
% ups copy -G "<ups_declare_options> <product> <version>" \
  [<options>] <product> <version>
```

23.4.2 Commonly Used Options

See section 23.4.3 *All Valid Options* for descriptions of each option.

`-f <flavor>` Or one of **`-0`**, **`-1`**, **`-2`**, **`-3`**, **`-4`**, or **`-H`** (alone or together with one of **`-0`**, **`-1`**, **`-2`**, **`-3`**, **`-4`**)

`-g <chainName>` Or one of **`-c`**, **`-d`**, **`-n`**, **`-o`**, **`-t`**

`-G "<options>"`

`-q <qualifierList>`

`-r <prodRootDir>`

`-T <path or URL>`

`-W`

`-X`

`-z <databaseList>`

Table 23.4.2-a:

-f <flavor>	Or one of **-0**, **-1**, **-2**, **-3**, **-4**, or **-H** (alone or together with one of **-0**, **-1**, **-2**, **-3**, **-4**)
-g <chainName>	Or one of **-c**, **-d**, **-n**, **-o**, **-t**
-G "<options>"	
-q <qualifierList>	
-r <prodRootDir>	
-T <path or URL>	
-W	
-X	
-z <databaseList>	

23.4.3 All Valid Options

-? (**"-?"** for **csh**) Prints command description and option usage information to screen

-c Finds source product instance chained to “current”

-d Finds source product instance chained to “development”

-f <flavor> Described below under “The flavor options”.

-g <chainName> Finds product instance chained to **<chainName>**

-G "<options>" Specifies options to be passed to the **ups declare** command for target product instance; see below

-H <flavor> Described below under “The flavor options”.

-m <tableFileName> Specifies table file name of source product instance

-M <tableFileDir> Specifies table file directory of source product instance

-n Finds source product instance chained to “new”

-o Finds source product instance chained to “old”

-O "<flags>" Sets the value of $UPS_OPTIONS to **<flags>**.

-q <qualifierList> Finds product instance on distribution node with the specified qualifiers (required and/or optional)

-r <prodRootDir> Specifies the product root directory of source product instance

-t Finds source product instance chained to “test”

-T <path or URL> Specifies archive file path or URL. This is used only for declarations in distribution databases for which products are maintained in tar or gzip (archived) format.

-U <upsDir> Specifies location of `ups` directory of source product instance; default value is **ups**

-v(**vvv**) Prints out extra debugging information.

-V Does not delete the temporary script files or partially installed products when command finishes; instead lists them on the screen

-W Uses environment variables (e.g., $SETUP_<PRODUCT>) to identify dependent product instances for target product (that is, it uses instances that are already setup in preference to what is listed in table file)

-X Executes the **ups declare** command instead of just echoing it

-z <databaseList> Specifies the database(s) in which to look for the product and its dependencies

-Z Times the command

Table 23.4.3-a:

-? ("-?" for csh)	Prints command description and option usage information to screen
-c	Finds source product instance chained to “current”
-d	Finds source product instance chained to “development”
-f <flavor>	Described below under “The flavor options”.
-g <chainName>	Finds product instance chained to **<chainName>**

Table 23.4.3-a:

-G "<options>"	Specifies options to be passed to the **ups declare** command for target product instance; see below
-H <flavor>	Described below under “The flavor options”.
-m <tableFileName>	Specifies table file name of source product instance
-M <tableFileDir>	Specifies table file directory of source product instance
-n	Finds source product instance chained to “new”
-o	Finds source product instance chained to “old”
-O "<flags>"	Sets the value of $UPS_OPTIONS to **<flags>**.
-q <qualifierList>	Finds product instance on distribution node with the specified qualifiers (required and/or optional)
-r <prodRootDir>	Specifies the product root directory of source product instance
-t	Finds source product instance chained to “test”
-T <path or URL>	Specifies archive file path or URL. This is used only for declarations in distribution databases for which products are maintained in tar or gzip (archived) format.
-U <upsDir>	Specifies location of **ups** directory of source product instance; default value is **ups**
-v(vvv)	Prints out extra debugging information.
-V	Does not delete the temporary script files or partially installed products when command finishes; instead lists them on the screen
-W	Uses environment variables (e.g., $SETUP_<PRODUCT>) to identify dependent product instances for target product (that is, it uses instances that are already setup in preference to what is listed in table file)
-X	Executes the **ups declare** command instead of just echoing it
-z <databaseList>	Specifies the database(s) in which to look for the product and its dependencies
-Z	Times the command

The flavor options

Flavor may be specified using **-f**, using **-H** by itself or in combination with any of **-0**, **-1**, **-2**, **-3**, **-4**, or just using one of **-0**, **-1**, **-2**, **-3**, **-4**. These options are not valid with each other (except **-H** with a number option).

-f <flavor> Finds source product instance of specified flavor, and sets target instance's flavor on distribution node, unless overridden in **-G**. Multiple values accepted, but **UPS** looks only at first in list.

-H <flavor> Specifies flavor and builds a flavor list for that family starting at the level specified. Multiple values accepted, but **UPS** looks only at first in list.

Can be used alone (without an accompanying number option). In this case, the best match is picked for source instance. This also determines target instance flavor unless overridden in **-G**.

If used with any of **-0**, **-1**, **-2**, **-3**, **-4**, instance with specified level of that flavor is picked as source instance; e.g., **-2H IRIX+6.2** is equivalent to **-f IRIX+6**.

-0 Specifies flavor as NULL; equivalent to **-f NULL**

-1 Specifies flavor as OS value up to the generic OS; e.g., on a SunOS machine it is equivalent to **-f SunOS**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX.

-2 Specifies flavor for product instance on local and distribution nodes as OS value up to the version; e.g., equivalent to **-f SunOS+5**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.

-3 Specifies flavor for product instance on local and distribution nodes upto the release of the version; e.g., equivalent to **-f SunOS+5.6**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.2.

-4 Specifies flavor for product instance on local and distribution nodes upto the patch number of the release; e.g., equivalent to **-f SunOS+5.6.2**; if given together with **-H IRIX+6.2.1**, flavor is then specified as IRIX+6.2.1.

Table 23.4.3-b:

-f <flavor>	Finds source product instance of specified flavor, and sets target instance's flavor on distribution node, unless overridden in **-G**. Multiple values accepted, but **UPS** looks only at first in list.

Table 23.4.3-b:

-H <flavor>	Specifies flavor and builds a flavor list for that family starting at the level specified. Multiple values accepted, but **UPS** looks only at first in list. Can be used alone (without an accompanying number option). In this case, the best match is picked for source instance. This also determines target instance flavor unless overridden in **-G**. If used with any of **-0**, **-1**, **-2**, **-3**, **-4**, instance with specified level of that flavor is picked as source instance; e.g., **-2H IRIX+6.2** is equivalent to **-f IRIX+6**.
-0	Specifies flavor as NULL; equivalent to **-f NULL**
-1	Specifies flavor as OS value up to the generic OS; e.g., on a SunOS machine it is equivalent to **-f SunOS**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX.
-2	Specifies flavor for product instance on local and distribution nodes as OS value up to the version; e.g., equivalent to **-f SunOS+5**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.
-3	Specifies flavor for product instance on local and distribution nodes upto the release of the version; e.g., equivalent to **-f SunOS+5.6**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.2.
-4	Specifies flavor for product instance on local and distribution nodes upto the patch number of the release; e.g., equivalent to **-f SunOS+5.6.1**; if given together with **-H IRIX+6.2.2**, flavor is then specified as IRIX+6.2.2.

23.4.4 Options Valid with -G

In order to distinguish the target product instance from the source, the declarations for the two instances must differ by at least one instance-identifying element. The **-G** option provides the means to specify the target instance identifiers; it takes a list of **ups declare** command line elements as an argument. Any identifier not specified via **-G** retains the value of the source instance. The elements valid for use with **-G** include **<product>**, **<version>** and the following subset of the **ups declare** options:

-A <nodeList>, **-c**, **-d**, **-D <origin>**, **-f <flavor>**, **-g <chainName>**, **-n**, **-o**, **-O "<flagList>"**, **-p "<description>"**, **-q <qualifierList>**, **-t**, **-z <databaseList>**, **-0**, **-1**, **-2**, **-3**, **-4**

See section 23.5 *ups declare* for details on each option. If the argument to **-G** includes the product version, the product name must be included ahead of the version; the first unflagged element is always interpreted as the product name and the second as the version.

23.4.5 More Detailed Description

The command **ups copy** is intended mainly for product developers declaring new instances on their development systems. It simplifies declaration of new instances of products that already exist in a **UPS** database. There is no restriction against using **ups copy** to copy the installation of a different product, however it's usually not particularly helpful in that situation.

Notes:

- **ups copy** runs **ups declare** if you use the **-X** (uppercase **-x**) option; if not used, the declare command is just echoed.
- Use the **-G** option to specify declaration information that is to be different from the installation you're using as a model. At least one instance-identifying element must be specified using **-G** to distinguish the source from the target instance.
- If you use the option **-W**, you will pick up the current environment. For example, if the previously declared instance depends on v1_0 of some product (e.g., **joe** v1_0), but the new instance should have **joe** v2_0 as a dependency, first run **setup joe v2_0**, then run **ups copy** with **-W**.

Internal Processes

- Process COPY action
- Create a table file entry for new instance (may use environment for **UPS** product requirements)
- If simulation only, write table file entry to temp and echo appropriate declare command
- Otherwise, write/merge in table file and declare new instance (see **ups declare** internals)
- Execute temp file

23.4.6 ups copy Examples

```
% ups copy dog v1 -G "dog v3 -f SunOS -q test -m v3.table -M
  ups\ -r /path/to/dog/v3"
```

This command runs a **ups copy** command for **dog** version v3, without the **-X** option so that the **ups declare** command just gets echoed, not executed. The **-G** argument here gives the product name and version plus options to be used by the **ups declare** command: the flavor (**-f SunOS**), a qualifier (**-q test**), the table file name and location (given via **-m** and **-M**), and the product root directory (given via **-r**). The command output looks like this:

```
/var/tmp/baaa006ni_table_dog
ups declare dog v3 -f "SunOS" -q "test" -r
"/path/to/dog/v3"\
 -U "ups" -m "v3.table" -M "ups"
```

The second command is identical to the first example except that the **-X** option instructs it to execute the **ups declare** command:

```
% ups copy dog v1 -G "dog v3 -f SunOS -q test -r /path/to/dog/v3
  \ -M ups -m v3.table" -X
```

23.5 ups declare

The **ups declare** command is used for two separate purposes:

1) to initially declare an instance to a database (and optionally add a chain at the same time)

2) to add a chain to a previously declared instance

23.5.1 Command Syntax

For initially declaring an instance

```
% ups declare <flavor_option> -r <prodRootDir>
  [<other_options>] \ <product> <version>
```

For declaring a chain

```
% ups declare <chain_option> [<other_options>] <product> \
  <version>
```

23.5.2 Commonly Used Options

See section 23.5.3 *All Valid Options* for descriptions of each option.

For initially declaring an instance

-f <flavor> Or one of **-0**, **-1**, **-2**, **-3**, **-4**, or **-H** (together with one of **-0**, **-1**, **-2**, **-3**, **-4**)

-g <chainName> Or one of **-c**, **-d**, **-n**, **-o**, **-t**

-m <tableFileName>

-q <qualifierList>

-r <prodRootDir>

-z <databaseList>

Table 23.5.2-a:

-f <flavor>	Or one of **-0**, **-1**, **-2**, **-3**, **-4**, or **-H** (together with one of **-0**, **-1**, **-2**, **-3**, **-4**)
-g <chainName>	Or one of **-c**, **-d**, **-n**, **-o**, **-t**
-m <tableFileName>	
-q <qualifierList>	
-r <prodRootDir>	
-z <databaseList>	

For declaring a chain

-f <flavor> Or one of **-0**, **-1**, **-2**, **-3**, **-4**, or **-H** (together with one of **-0**, **-1**, **-2**, **-3**, **-4**)

-g <chainName> Or one of **-c**, **-d**, **-n**, **-o**, **-t**

-q <qualifierList>

-z <databaseList>

Table 23.5.2-b:

-f <flavor>	Or one of **-0**, **-1**, **-2**, **-3**, **-4**, or **-H** (together with one of **-0**, **-1**, **-2**, **-3**, **-4**)
-g <chainName>	Or one of **-c**, **-d**, **-n**, **-o**, **-t**
-q <qualifierList>	
-z <databaseList>	

23.5.3 All Valid Options

Valid only for initially declaring an instance (not for assigning a chain)

`-A <nodeList>` Specifies nodes authorized to access the product; sets the keyword AUTHORIZED_NODES

`-b <compileFile>` Specifies name of the output file for the **`ups compile`** command (described in Chapter 38: *Use of Compile Scripts in Table Files*); sets the keyword COMPILE_FILE

`-D "<origin>"` Specifies the product's master source file; sets the keyword ORIGIN (all spaces get removed from **`<origin>`** for the keyword value)

`-L` Adds the STATISTICS keyword to the version file, thereby instructing **UPS** to keep statistics on this product instance. A record of the form:

```
"tcl" "v7_3q" "IRIX" "" "" "berman"
"1998-03-13 17.56.54 GMT" "list"
```

will get added to the file `$PRODUCTS/.upsfiles/statistics/<product>` each time a **UPS** command is run on this instance.

`-u <compileDir>` Specifies the directory for the output file (which is named via the **`-b`** option) for the **`ups compile`** command; sets the keyword COMPILE_DIR.

Table 23.5.3-a:

`-A <nodeList>`	Specifies nodes authorized to access the product; sets the keyword AUTHORIZED_NODES
`-b <compileFile>`	Specifies name of the output file for the **`ups compile`** command (described in Chapter 38: *Use of Compile Scripts in Table Files*); sets the keyword COMPILE_FILE
`-D "<origin>"`	Specifies the product's master source file; sets the keyword ORIGIN (all spaces get removed from **`<origin>`** for the keyword value)
`-L`	Adds the STATISTICS keyword to the version file, thereby instructing **UPS** to keep statistics on this product instance. A record of the form: **`"tcl" "v7_3q" "IRIX" "" "" "berman" "1998-03-13 17.56.54 GMT" "list"`** will get added to the file `$PRODUCTS/.upsfiles/statistics/<product>` each time a **UPS** command is run on this instance.

Table 23.5.3-a:

-u <compileDir>	Specifies the directory for the output file (which is named via the **-b** option) for the **ups compile** command; sets the keyword COMPILE_DIR.

Valid for both functions

-? (**"-?"** for **csh**) Prints command description and option usage information to screen.

-c Chains the product instance to "current"; when this chain gets declared, the corresponding ACTION=CURRENT in the table file gets executed, if it exists.

-C When initially declaring a product, **-C** prevents execution of the CONFIGURE action.

When declaring a chain, **-C** prevents execution of the corresponding chain action.

-d Chains the product instance to "development"; when this chain gets declared, the corresponding ACTION=DEVELOPMENT in the table file gets executed, if it exists.

-f <flavor> Described below under "The flavor options".

-g <chainName> Chains the product instance to **<chainName>** (this is useful for user-defined chains). When any chain gets declared, the corresponding ACTION=<CHAIN_NAME> in the table file gets executed.

-H <flavor> Described below under "The flavor options".

-m <tableFileName> Specifies table file name; when initially declaring a product, sets the keyword TABLE_FILE.

-M <tableFileDir> Specifies table file directory; when initially declaring a product, sets the keyword TABLE_DIR. Specify only if file is not in one of the two default locations, namely under `$PRODUCTS/<product>` or in the `ups` directory.

-n Chains the product instance to "new"; when this chain gets declared, the corresponding ACTION=NEW in the table file gets executed, if it exists.

-o Chains the product instance to "old"; when this chain gets declared, the corresponding ACTION=OLD in the table file gets executed, if it exists.

-O "<flags>" Sets the value of $UPS_OPTIONS to **<flags>**.

-q <qualifiers> When initially declaring a product, **-q** specifies required and/or optional qualifiers to include in the declaration, and sets the keyword QUALIFIERS.

When adding a chain, **-q** specifies required and/or optional qualifiers to identify the instance.

-r <prodRootDir> Specifies the product root directory; when initially declaring a product, sets the keyword PROD_DIR.

A note for developers: you may find it convenient to use the construction **-r \pwd** if you're working in the product root directory.

-t Chains the product instance to "test"; when this chain gets declared, the corresponding ACTION=TEST in the table file gets executed, if it exists.

-T <path or URL> Specifies archive file path or URL. This is used only for declarations in distribution databases for which products are maintained in tar or gzip (archived) format. When initially declaring a product, it sets the keyword ARCHIVE_FILE.

-U <upsDir> Specifies location of `ups` directory; default value is **ups**; when initially declaring a product, sets the keyword UPS_DIR

-v(**vvv**) Prints out extra debugging information.

-V Does not delete the temporary script files or partially installed products when command finishes; instead lists them on the screen

-z <databaseList> Specifies the database in which to declare the product (see section 27.1 *Database Selection Algorithm*); or, if adding a chain, specifies the database(s) in which product is declared

-Z Times the command

Table 23.5.3-b:

-? ("-?" for csh)	Prints command description and option usage information to screen.
-c	Chains the product instance to "current"; when this chain gets declared, the corresponding ACTION=CURRENT in the table file gets executed, if it exists.
-C	When initially declaring a product, **-C** prevents execution of the CONFIGURE action. When declaring a chain, **-C** prevents execution of the corresponding chain action.

Table 23.5.3-b:

-d	Chains the product instance to "development"; when this chain gets declared, the corresponding ACTION=DEVELOPMENT in the table file gets executed, if it exists.
-f <flavor>	Described below under "The flavor options".
-g <chainName>	Chains the product instance to **<chainName>** (this is useful for user-defined chains). When any chain gets declared, the corresponding ACTION=<CHAIN_NAME> in the table file gets executed.
-H <flavor>	Described below under "The flavor options".
-m <tableFileName>	Specifies table file name; when initially declaring a product, sets the keyword TABLE_FILE.
-M <tableFileDir>	Specifies table file directory; when initially declaring a product, sets the keyword TABLE_DIR. Specify only if file is not in one of the two default locations, namely under `$PRODUCTS/<product>` or in the `ups` directory.
-n	Chains the product instance to "new"; when this chain gets declared, the corresponding ACTION=NEW in the table file gets executed, if it exists.
-o	Chains the product instance to "old"; when this chain gets declared, the corresponding ACTION=OLD in the table file gets executed, if it exists.
-O "<flags>"	Sets the value of $UPS_OPTIONS to **<flags>**.
-q <qualifiers>	When initially declaring a product, **-q** specifies required and/or optional qualifiers to include in the declaration, and sets the keyword QUALIFIERS. When adding a chain, **-q** specifies required and/or optional qualifiers to identify the instance.
-r <prodRootDir>	Specifies the product root directory; when initially declaring a product, sets the keyword PROD_DIR. A note for developers: you may find it convenient to use the construction **-r \pwd** if you're working in the product root directory.
-t	Chains the product instance to "test"; when this chain gets declared, the corresponding ACTION=TEST in the table file gets executed, if it exists.
-T <path or URL>	Specifies archive file path or URL. This is used only for declarations in distribution databases for which products are maintained in tar or gzip (archived) format. When initially declaring a product, it sets the keyword ARCHIVE_FILE.
-U <upsDir>	Specifies location of `ups` directory; default value is **ups**; when initially declaring a product, sets the keyword UPS_DIR
-v(vvv)	Prints out extra debugging information.
-V	Does not delete the temporary script files or partially installed products when command finishes; instead lists them on the screen

Table 23.5.3-b:

-z <dataBaseList>	Specifies the database in which to declare the product (see section 27.1 *Database Selection Algorithm*); or, if adding a chain, specifies the database(s) in which product is declared
-Z	Times the command

The flavor options

Flavor may be specified using **-f**, or using **-H** in combination with any of **-0**, **-1**, **-2**, **-3**, **-4**, or just using one of **-0**, **-1**, **-2**, **-3**, **-4**. These options are not valid with each other (except **-H** with a number option).

-f <flavor> Declares product instance as specified flavor; when initially declaring a product, sets the keyword FLAVOR.

-H <flavor> Must be used with any of **-0**, **-1**, **-2**, **-3**, **-4**. Specifies flavor and builds a flavor list for that family starting at the level specified. **UPS** finds the product instance of specified level of that flavor; e.g., **-2H IRIX+6.2** is equivalent to **-f IRIX+6**.

-0 Specifies flavor as NULL; equivalent to **-f NULL**

-1 Specifies flavor as OS value up to the generic OS; e.g., on a SunOS machine it is equivalent to **-f SunOS**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX.

-2 Specifies flavor for product instance on local and distribution nodes as OS value up to the version; e.g., equivalent to **-f SunOS+5**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.

-3 Specifies flavor for product instance on local and distribution nodes upto the release of the version; e.g., equivalent to **-f SunOS+5.6**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.2.

-4 Specifies flavor for product instance on local and distribution nodes upto the patch number of the release; e.g., equivalent to **-f SunOS+5.6.2**; if given together with **-H IRIX+6.2.1**, flavor is then specified as IRIX+6.2.1.

Table 23.5.3-c:

-f <flavor>	Declares product instance as specified flavor; when initially declaring a product, sets the keyword FLAVOR.

Table 23.5.3-c:

-H <flavor>	Must be used with any of **-0**, **-1**, **-2**, **-3**, **-4**. Specifies flavor and builds a flavor list for that family starting at the level specified. **UPS** finds the product instance of specified level of that flavor; e.g., **-2H IRIX+6.2** is equivalent to **-f IRIX+6**.
-0	Specifies flavor as NULL; equivalent to **-f NULL**
-1	Specifies flavor as OS value up to the generic OS; e.g., on a SunOS machine it is equivalent to **-f SunOS**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX.
-2	Specifies flavor for product instance on local and distribution nodes as OS value up to the version; e.g., equivalent to **-f SunOS+5**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.
-3	Specifies flavor for product instance on local and distribution nodes upto the release of the version; e.g., equivalent to **-f SunOS+5.6**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.2.
-4	Specifies flavor for product instance on local and distribution nodes upto the patch number of the release; e.g., equivalent to **-f SunOS+5.6.2**; if given together with **-H IRIX+6.2.1**, flavor is then specified as IRIX+6.2.1.

23.5.4 More Detailed Description

Declaring an Instance for the First Time

In the **ups declare** command:

- **-C** prevents execution of the CONFIGURE action, if any, and is normally not included. The default (and usually desired) behavior is to execute the CONFIGURE action.
- You must include a flavor specification, there is no default flavor when a product is first declared. It sets the value of the FLAVOR keyword.
- Most products require a table file. The table file must exist before running **ups declare**! In most cases you need to include the table file name (**-m**). (Since in a very few cases a table file isn't required, **-m** is not a strictly required option.)
- If the product's table file was placed in either of the two default locations (under `/path/to/database/<product>` or in the product's `ups` directory), then **-M <table_file_dir>** is not needed. Only use the **-M** option if you have moved the table file to a separate location where **UPS** won't otherwise find it.
- In most cases you need to include product root directory (**-r**). Exceptions include wrapper products which consist only of a table file, and thus have no product root directory.

- If the product's `ups` directory tar file was unwound in the default location (`$<PRODUCT>_DIR/ups`), then **`-U <ups_dir>`** is not needed. If the `ups` directory is located elsewhere (or named differently), this specification must be included. In general, you should not include this qualifier.

- If you choose not to specify the target database explicitly (**`-z <database>`**), **UPS** chooses it automatically via the $PRODUCTS variable. If $PRODUCTS points to multiple databases, you need to be a little careful about database selection. The database matching algorithm is described in section 27.1 *Database Selection Algorithm*.

  If the product has dependencies that are declared in different databases, **UPS** must be able to find all of them in order to resolve the dependencies. You can rely on $PRODUCTS if all the necessary databases are included in it. Otherwise specify them on the command line (e.g., **`-z <database1>:<database2>:...>`** or **`-z $PRODUCTS:<database1>:<database2>:...`**).
- You may include chain information on this command. See the description below.

Adding Chains to an Existing Instance

When you add one or more chains to an existing instance, **UPS** doesn't allow you to change anything else about that instance. Regarding the options to specify:

- You of course need to specify the chain or chains to add using either **`-g <chain_name>`** or one of **`-c`**, **`-d`**, **`-n`**, **`-o`**, **`-t`**.
- **`-C`** prevents the corresponding chain action(s) in the table file from being executed.
- You do not strictly need to specify flavor. **UPS** will default to the best flavor match (described in section 27.2.4 *Flavor and Qualifier Matching Algorithm*). You can override the default using **`-f`** or one of the number options (**`-0`**, **`-1`**, **`-2`**, **`-3`**, **`-4`**).
- Specify qualifiers (**`-q`**) as necessary to select the appropriate instance.
- Specify the database (**`-z`**) as necessary to select the appropriate instance.

Internal Processes

- Find database to use
- If necessary, process 'UNCHAIN' action
- Process DECLARE action
- If necessary, process CONFIGURE action
- If necessary, warn if there is a TAILOR action
- If necessary, process the 'CHAIN' action
- If necessary, warn if there are START/STOP actions
- If current chain, try to copyman, copycatman and copyInfo files
- Execute the temp file

- If successful, modify all appropriate files on disk

23.5.5 ups declare Examples

Declare a product with no chain using defaults where possible

```
% ups declare myprod v1_0 -f Linux+2 -m myprod.table \
  -r /path/to/myprod/v1_0/Linux+2
```

UPS finds the product in the directory given by the **-r** option, and declares it to a database in $PRODUCTS according to the selection algorithm discussed in 27.1 *Database Selection Algorithm*. The product instance gets declared with the specified name, version and flavor, and no qualifiers. **UPS** looks for the table file, called `myprod.table` in the two default locations (command fails if table file doesn't exist, or is not found in either location).

Declare a product with no chain using defaults where possible

```
% ups declare myprod2 v1_0 -2 -g test -m myprod2.table \
  -r myprod2/v1_0/IRIX+6 -z /my/local/db:$PRODUCTS
```

For this example, we assume the local machine flavor is IRIX+6.2. **UPS** finds the product in the directory given by the **-r** option. The specified path is taken relative to PROD_DIR_PREFIX. **UPS** declares the product to one of the listed databases according to the selection algorithm discussed in 27.1 *Database Selection Algorithm*. Each of the dependencies, if any, must exist in at least one of the listed databases.

The product instance gets declared with the specified name, version, no qualifiers, and the chain "test". The flavor declaration is the level **-2** specification of the machine, namely IRIX+6. **UPS** looks for the table file, called `myprod2.table` in the two default locations (command fails if table file doesn't exist, or is not found in either location).

Add a chain to a previously declared instance

```
% ups declare myprod2 v1_0 -2 -g test -m myprod2.table \
  -r myprod2/v1_0/IRIX+6 -z /my/local/db:$PRODUCTS

% ups declare myprod2 v1_0 -2 -c -z /my/local/db:$PRODUCTS
```

This command declares the product instance of the previous example "current" (via the **-c** option). Generally, a product is first declared as "test", and then after a "debugging period" (often several weeks), an updated release is cut and chained to "current". Notes:

- In most **UPS/UPD** commands you specify either chain (often defaulted to "current") or version. Here you need both: the version is required to identify the instance, and the chain is required because it is being assigned.

- You should specify the database **`/my/local/db`** first since that's how it was initially declared. **UPS** will traverse the databases in the same order to find the right instance.

23.6 ups depend

The **`ups depend`** command lists product dependencies of the specified product instance(s) as declared in the (local) database. On user nodes it is generally used to determine what products will get setup along with the "parent" product. **UPD** uses it on product servers to determine what dependencies to install.

23.6.1 Command Syntax

```
% ups depend [<options>] <product> [<version>]
```

23.6.2 Commonly Used Options

See section 23.6.3 *All Valid Options* for descriptions of each option.

`-f <flavor>` Or one of **`-0`**, **`-1`**, **`-2`**, **`-3`**, **`-4`**, or **`-H`** (alone or together with one of **`-0`**, **`-1`**, **`-2`**, **`-3`**, **`-4`**)

`-g <chainName>` Or one of **`-c`**, **`-d`**, **`-n`**, **`-o`**, **`-t`**

`-j`

`-K <keywordList>`

`-l`

`-q <qualifierList>`

`-R`

`-z <databaseList>`

Table 23.6.2-a:

`-f <flavor>`	Or one of **`-0`**, **`-1`**, **`-2`**, **`-3`**, **`-4`**, or **`-H`** (alone or together with one of **`-0`**, **`-1`**, **`-2`**, **`-3`**, **`-4`**)
`-g <chainName>`	Or one of **`-c`**, **`-d`**, **`-n`**, **`-o`**, **`-t`**
`-j`	

Table 23.6.2-a:

-K <key-wordList>	
-l	
-q <qual-ifier-List>	
-R	
-z <data-baseList>	

23.6.3 All Valid Options

-? (**"-?"** for **csh**) Prints command description and option usage information to screen

-c Finds product instance chained to “current”

-d Finds product instance chained to “development”

-f <flavor> Described below under “The flavor options”.

-g <chainName> Finds product instance chained to **<chainName>**

-H <flavor> Described below under “The flavor options”.

-j Ignores lower level dependencies, finds direct dependencies of top level product only

-K <keywordList> Returns values of specified keywords only; valid keywords are listed in section 28.4 *List of Supported Keywords*

-l Produces a long listing including all the table file functions that would be executed in a **setup** command.

-m <tableFileName> Specifies table file name

-M <tableFileDir> Specifies table file directory

-n Finds product instance chained to “new”

-o Finds product instance chained to “old”

-q <qualifierList> Finds product instance with the specified qualifiers (required and/or optional)

-r <prodRootDir> Specifies the product root directory

-R Lists only the required (non-optional) dependencies.

-t	Finds product instance chained to “test”
-U <upsDir>	Specifies location of `ups` directory; default value is **ups**
-v(**vvv**)	Prints out extra debugging information.
-V	Does not delete the temporary script files or partially installed products when command finishes; instead lists them on the screen
-z <databases>	Specifies the database(s) to search
-Z	Times the command

Table 23.6.3-a:

-? ("-?" for csh)	Prints command description and option usage information to screen
-c	Finds product instance chained to “current”
-d	Finds product instance chained to “development”
-f <flavor>	Described below under “The flavor options”.
-g <chainName>	Finds product instance chained to **<chainName>**
-H <flavor>	Described below under “The flavor options”.
-j	Ignores lower level dependencies, finds direct dependencies of top level product only
-K <keywordList>	Returns values of specified keywords only; valid keywords are listed in section 28.4 *List of Supported Keywords*
-l	Produces a long listing including all the table file functions that would be executed in a **setup** command.
-m <tableFileName>	Specifies table file name
-M <tableFileDir>	Specifies table file directory
-n	Finds product instance chained to “new”
-o	Finds product instance chained to “old”

Table 23.6.3-a:

-q <qualifier-List>	Finds product instance with the specified qualifiers (required and/or optional)
-r <prodRootDir>	Specifies the product root directory
-R	Lists only the required (non-optional) dependencies.
-t	Finds product instance chained to "test"
-U <upsDir>	Specifies location of `ups` directory; default value is **`ups`**
-v(vvv)	Prints out extra debugging information.
-V	Does not delete the temporary script files or partially installed products when command finishes; instead lists them on the screen
-z <databases>	Specifies the database(s) to search
-Z	Times the command

The flavor options

Flavor may be specified using **`-f`**, using **`-H`** by itself or in combination with any of **`-0`**, **`-1`**, **`-2`**, **`-3`**, **`-4`**, or just using one of **`-0`**, **`-1`**, **`-2`**, **`-3`**, **`-4`**. These options are not valid with each other (except **`-H`** with a number option).

`-f <flavor>` Finds product instance of specified flavor. If specified and no exact match is found, the command fails. Multiple values accepted, but **UPS** looks only at first in list.

`-H <flavor>` Specifies flavor and builds a flavor list for that family starting at the level specified. Multiple values accepted, but **UPS** looks only at first in list.

Can be used alone (without an accompanying number option). In this case, **UPS** finds the best match instance for the specified flavor family.

If used with any of **`-0`**, **`-1`**, **`-2`**, **`-3`**, **`-4`**, **UPS** finds the product instance of specified level of that flavor; e.g., **`-2H IRIX+6.2`** is equivalent to **`-f IRIX+6`**.

`-0` Specifies flavor as NULL; equivalent to **`-f NULL`**

`-1` Specifies flavor as OS value up to the generic OS; e.g., on a SunOS machine it is equivalent to **`-f SunOS`**; if given together with **`-H IRIX+6.2`**, flavor is then specified as IRIX.

-2 Specifies flavor for product instance on local and distribution nodes as OS value up to the version; e.g., equivalent to **-f SunOS+5**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.

-3 Specifies flavor for product instance on local and distribution nodes upto the release of the version; e.g., equivalent to **-f SunOS+5.6**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.2.

-4 Specifies flavor for product instance on local and distribution nodes upto the patch number of the release; e.g., equivalent to **-f SunOS+5.6.2**; if given together with **-H IRIX+6.2.1**, flavor is then specified as IRIX+6.2.1.

Table 23.6.3-b:

-f <flavor>	Finds product instance of specified flavor. If specified and no exact match is found, the command fails. Multiple values accepted, but **UPS** looks only at first in list.
-H <flavor>	Specifies flavor and builds a flavor list for that family starting at the level specified. Multiple values accepted, but **UPS** looks only at first in list. Can be used alone (without an accompanying number option). In this case, **UPS** finds the best match instance for the specified flavor family. If used with any of **-0**, **-1**, **-2**, **-3**, **-4**, **UPS** finds the product instance of specified level of that flavor; e.g., **-2H IRIX+6.2** is equivalent to **-f IRIX+6**.
-0	Specifies flavor as NULL; equivalent to **-f NULL**
-1	Specifies flavor as OS value up to the generic OS; e.g., on a SunOS machine it is equivalent to **-f SunOS**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX.
-2	Specifies flavor for product instance on local and distribution nodes as OS value up to the version; e.g., equivalent to **-f SunOS+5**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.
-3	Specifies flavor for product instance on local and distribution nodes upto the release of the version; e.g., equivalent to **-f SunOS+5.6**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.2.
-4	Specifies flavor for product instance on local and distribution nodes upto the patch number of the release; e.g., equivalent to **-f SunOS+5.6.2**; if given together with **-H IRIX+6.2.1**, flavor is then specified as IRIX+6.2.1.

23.6.4 ups depend Examples

Execute command using all default values (no options)

```
% ups depend exmh
```

The first example requests information for the default instance of **exmh**, which is the one chained to "current" for the best-match flavor of the machine (SunOS+5 for this example). The output looks like this:

```
exmh v2_0_2 -f NULL -z /afs/fnal.gov/ups/db -g current
|__expect v5_25 -f SunOS+5 -z /afs/fnal.gov/ups/db -g
current
|  |__tk v8_0_2 -f SunOS+5 -z /afs/fnal.gov/ups/db
|     |__tcl v8_0_2 -f SunOS+5 -z /afs/fnal.gov/ups/db
|__mh v6_8_3c -f SunOS+5 -z /afs/fnal.gov/ups/db -g current
|   |__mailtools v2_3 -f NULL -z /afs/fnal.gov/ups/db -g
current
|__mimetools v2_7a -f SunOS+5 -z /afs/fnal.gov/ups/db -g
current
|__glimpse v3_0a -f SunOS+5 -z /afs/fnal.gov/ups/db -g
current
|__www v3_0 -f NULL -z /afs/fnal.gov/ups/db -g current
|   |__netscape v4_05 -f SunOS+5 -z /afs/fnal.gov/ups/db -g
current
|  |  |__ghostview v4_0 -f SunOS+5 -z /afs/fnal.gov/ups/db -g
current
|  |  |__ximagetools v4_0 -f NULL -z /afs/fnal.gov/ups/db -g
current
|  |  |  |__imagelibs v1_0 -f SunOS+5 -z /afs/fnal.gov/ups/db
|     |     |     |__imagemagick v4_04 -f SunOS+5 -z
/afs/fnal.gov/ups/db
|  |  |  |__xfig v3_20 -f SunOS+5 -z /afs/fnal.gov/ups/db
|  |  |  |__xanim v2_70_64 -f SunOS+5 -z /afs/fnal.gov/ups/db
|   |   |__xpdf v0_7 -f SunOS+5 -z /afs/fnal.gov/ups/db -g
current
|   |__lynx v2_8_1 -f SunOS+5 -z /afs/fnal.gov/ups/db -g
current
|__ispell v3_1b -f SunOS+5 -z /afs/fnal.gov/ups/db -g
current
```

If a chain rather than a version number is used to specify the instance (as is the case for the default "current" instance), then the chain appears in the output line for the product (notice the `-g current` in the first line); otherwise the chain is not listed (compare to the following example). Compare this example to the following one:

```
% ups depend exmh v2_0_2
```

The instance is the same, the difference is that it was specified using the version rather than the chain (output edited for brevity):

```
exmh v2_0_2 -f NULL -z /afs/fnal.gov/ups/db
|__expect  v5_25  -f  SunOS+5  -z  /afs/fnal.gov/ups/db  -g
current
|  |__tk v8_0_2 -f SunOS+5 -z /afs/fnal.gov/ups/db
|     |__tcl v8_0_2 -f SunOS+5 -z /afs/fnal.gov/ups/db
|__mh v6_8_3c -f SunOS+5 -z /afs/fnal.gov/ups/db -g current
...
```

List only required dependencies

```
% ups depend -R exmh
```

We use the **-R** option to request only the dependencies listed as “required” for the same product instance as in the previous examples:

```
exmh v2_0_2 -f NULL -z /afs/fnal.gov/ups/db -g current
|__expect v5_25 -f SunOS+5 -z /afs/fnal.gov/ups/db -g
current
|  |__tk v8_0_2 -f SunOS+5 -z /afs/fnal.gov/ups/db
|     |__tcl v8_0_2 -f SunOS+5 -z /afs/fnal.gov/ups/db
|__mh v6_8_3c -f SunOS+5 -z /afs/fnal.gov/ups/db -g current
|   |__mailtools v2_3 -f NULL -z /afs/fnal.gov/ups/db -g
current
|__mimetools v2_7a -f SunOS+5 -z /afs/fnal.gov/ups/db -g
current
```

List a subset of fields for required dependencies only

```
% ups depend -RK product:version:flavor exmh
```

To return a subset of the output fields, we include the **-K** option (described in section 25.2.3 -*K*):

```
"exmh" "v2_0_2" "NULL"
"expect" "v5_25" "SunOS+5"
"tk" "v8_0_2" "SunOS+5"
"tcl" "v8_0_2" "SunOS+5"
"mh" "v6_8_3c" "SunOS+5"
"mailtools" "v2_3" "NULL"
"mimetools" "v2_7a" "SunOS+5"
```

List only direct dependencies

```
% ups depend -j exmh
```

We use the **-j** option to request "just" the direct dependencies of the specified product (dependencies of dependencies are omitted):

```
exmh v2_0_2 -f NULL -z /afs/fnal.gov/ups/db -g current
|__expect v5_25 -f SunOS+5 -z /afs/fnal.gov/ups/db -g
current
|__mh v6_8_3c -f SunOS+5 -z /afs/fnal.gov/ups/db -g current
|__mimetools v2_7a -f SunOS+5 -z /afs/fnal.gov/ups/db -g
current
|__glimpse v3_0a -f SunOS+5 -z /afs/fnal.gov/ups/db -g
current
|__www v3_0 -f NULL -z /afs/fnal.gov/ups/db -g current
|__netscape v4_05 -f SunOS+5 -z /afs/fnal.gov/ups/db -g
current
|__ispell v3_1b -f SunOS+5 -z /afs/fnal.gov/ups/db -g
current
```

23.7 ups exist

The **ups exist** command is used to test whether a **setup** command issued with the same command line elements is likely to succeed. It was designed primarily for use in scripts.

23.7.1 Command Syntax

```
% ups exist [<options>] <product> [<version>]
```

23.7.2 Commonly Used Options

See section 23.7.3 *All Valid Options* for descriptions of each option.

-f <flavor> Or one of **-0**, **-1**, **-2**, **-3**, **-4**, or **-H** (alone or together with one of **-0**, **-1**, **-2**, **-3**, **-4**)

-g <chainName> Or one of **-c**, **-d**, **-n**, **-o**, **-t**

-j

-q <qualifierList>

-z <databaseList>

Table 23.7.2-a:

-f <flavor>	Or one of **-0**, **-1**, **-2**, **-3**, **-4**, or **-H** (alone or together with one of **-0**, **-1**, **-2**, **-3**, **-4**)
-g <chainName>	Or one of **-c**, **-d**, **-n**, **-o**, **-t**
-j	
-q <qualifierList>	
-z <databaseList>	

23.7.3 All Valid Options

-? (**"-?"** for **csh**) Prints command description and option usage information to screen

-B <depProdName>= "<options>"

Specifies options to prepend to the `setupRequired` line (in table file) for the dependent product **`<depProdName>`**

`-c` Finds product instance chained to “current”

`-d` Finds product instance chained to “development”

`-e` Sets $UPS_EXTENDED (to the value **`1`**).

`-f <flavor>` Described below under “The flavor options”.

`-g <chainName>` Finds product instance chained to **`<chainName>`**

`-H <flavor>` Described below under “The flavor options”.

`-j` Ignores dependencies, operates just on top level product

`-k` Prevents execution of unsetup files prior to (subsequent) setup

`-m <tableFileName>` Specifies table file name

`-M <tableFileDir>` Specifies table file directory

`-n` Finds product instance chained to “new”

`-o` Finds product instance chained to “old”

`-O "<flags>"` Sets the value of $UPS_OPTIONS to **`<flags>`**.

`-P` Requires **UPS** to rely only on information supplied on the command line to locate the product instance (prevents **UPS** from searching in a database)

`-q <qualifierList>` Finds product instance with the specified qualifiers (required and/or optional)

`-r <prodRootDir>` Specifies the product root directory

`-t` Finds product instance chained to “test”

`-U <upsDir>` Specifies location of `ups` directory; default value is **`ups`**

`-v`(**`vvv`**) Prints out extra debugging information.

`-V` Does not delete the temporary script files or partially installed products when command finishes; instead lists them on the screen

`-z <databaseList>` Specifies the database(s) in which to look for the product and its dependencies

`-Z` Times the command

Table 23.7.3-a:

-? ("-?" for csh)	Prints command description and option usage information to screen
-B <dep-Prod-Name>= "<options>"	Specifies options to prepend to the `setupRequired` line (in table file) for the dependent product **<depProdName>**
-c	Finds product instance chained to "current"
-d	Finds product instance chained to "development"
-e	Sets $UPS_EXTENDED (to the value **1**).
-f <flavor>	Described below under "The flavor options".
-g <chainName>	Finds product instance chained to **<chainName>**
-H <flavor>	Described below under "The flavor options".
-j	Ignores dependencies, operates just on top level product
-k	Prevents execution of unsetup files prior to (subsequent) setup
-m <table-FileName>	Specifies table file name
-M <table-FileDir>	Specifies table file directory
-n	Finds product instance chained to "new"
-o	Finds product instance chained to "old"
-O "<flags>"	Sets the value of $UPS_OPTIONS to **<flags>**.
-P	Requires **UPS** to rely only on information supplied on the command line to locate the product instance (prevents **UPS** from searching in a database)
-q <qualifier-List>	Finds product instance with the specified qualifiers (required and/or optional)
-r <prodRootDir>	Specifies the product root directory
-t	Finds product instance chained to "test"

Table 23.7.3-a:

-U <ups-Dir>	Specifies location of `ups` directory; default value is **ups**
-v(vvv)	Prints out extra debugging information.
-V	Does not delete the temporary script files or partially installed products when command finishes; instead lists them on the screen
-z <databaseList>	Specifies the database(s) in which to look for the product and its dependencies
-Z	Times the command

The flavor options

Flavor may be specified using **-f**, using **-H** by itself or in combination with any of **-0**, **-1**, **-2**, **-3**, **-4**, or just using one of **-0**, **-1**, **-2**, **-3**, **-4**. These options are not valid with each other (except **-H** with a number option).

-f <flavor> Finds product instance of specified flavor. If specified and no exact match is found, the command fails. Multiple values accepted, but **UPS** looks only at first in list.

-H <flavor> Specifies flavor and builds a flavor list for that family starting at the level specified. Multiple values accepted, but **UPS** looks only at first in list.

Can be used alone (without an accompanying number option). In this case, **UPS** finds the best match instance for the specified flavor family.

If used with any of **-0**, **-1**, **-2**, **-3**, **-4**, **UPS** finds the product instance of specified level of that flavor; e.g., **-2H IRIX+6.2** is equivalent to **-f IRIX+6**.

-0 Specifies flavor as NULL; equivalent to **-f NULL**

-1 Specifies flavor as OS value up to the generic OS; e.g., on a SunOS machine it is equivalent to **-f SunOS**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX.

-2 Specifies flavor for product instance on local and distribution nodes as OS value up to the version; e.g., equivalent to **-f SunOS+5**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.

-3 Specifies flavor for product instance on local and distribution nodes upto the release of the version; e.g., equivalent to **-f SunOS+5.6**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.2.

-4	Specifies flavor for product instance on local and distribution nodes upto the patch number of the release; e.g., equivalent to **-f SunOS+5.6.2**; if given together with **-H IRIX+6.2.1**, flavor is then specified as IRIX+6.2.1.

Table 23.7.3-b:

-f <flavor>	Finds product instance of specified flavor. If specified and no exact match is found, the command fails. Multiple values accepted, but **UPS** looks only at first in list.
-H <flavor>	Specifies flavor and builds a flavor list for that family starting at the level specified. Multiple values accepted, but **UPS** looks only at first in list. Can be used alone (without an accompanying number option). In this case, **UPS** finds the best match instance for the specified flavor family. If used with any of **-0**, **-1**, **-2**, **-3**, **-4**, **UPS** finds the product instance of specified level of that flavor; e.g., **-2H IRIX+6.2** is equivalent to **-f IRIX+6**.
-0	Specifies flavor as NULL; equivalent to **-f NULL**
-1	Specifies flavor as OS value up to the generic OS; e.g., on a SunOS machine it is equivalent to **-f SunOS**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX.
-2	Specifies flavor for product instance on local and distribution nodes as OS value up to the version; e.g., equivalent to **-f SunOS+5**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.
-3	Specifies flavor for product instance on local and distribution nodes upto the release of the version; e.g., equivalent to **-f SunOS+5.6**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.2.
-4	Specifies flavor for product instance on local and distribution nodes upto the patch number of the release; e.g., equivalent to **-f SunOS+5.6.2**; if given together with **-H IRIX+6.2.1**, flavor is then specified as IRIX+6.2.1.

23.7.4 More Detailed Description

The **ups exist** command is used to test whether a **setup** command issued with the same command line elements is likely to succeed. As for all the **UPS/UPD** commands, if the **setup** command finds a corresponding action in the selected table file, it

1) translates the functions listed under the action into shell commands,

2) writes them to a temporary script in $TMPDIR (if $TMPDIR isn't set, the default is `/tmp`), and

3) invokes the script to execute the shell commands.

ups exist checks for a properly declared matching instance, and verifies that you have the necessary permissions to create the temporary script. If so, it creates the script, but it does not execute it.

In the C shell family **ups exist** sets the $status variable to `0` if it was able to create the temporary file, or to `1` for error. In the Bourne shell family, it sets the $? variable similarly.

This command is rarely used from the command line, and is more useful in scripts where a failed setup could cause the script to abort. When issued from the command line, it returns no output if the command succeeds.

Internal Processes

- Check node authorization
- If necessary, process UNSETUP action
- Simulate SETUP action

23.7.5 ups exist Examples

This command is rarely used from the command line, and is more useful in scripts where a failed setup could cause the script to abort. When issued from the command line, it returns no output if the command succeeds.

In the C shell family **ups exist** sets the $status variable to `0` if it was able to create the temporary file, or to `1` for error. In the Bourne shell family, it sets the $? variable similarly. As an example, we can run **ups list** and find that there is a current instance of the product **tex** for the flavor IRIX+6 but not for IRIX+6.2. Running **ups exist** for each flavor, we see that the variables get set accordingly. For the C shell family:

```
% ups exist tex -f IRIX+6; echo $status
   0
% ups exist tex -f IRIX+6.2; echo $status
   1
```

For the Bourne shell family:

```
$ ups exist tex -f IRIX+6; echo $?
   0
$ ups exist tex -f IRIX+6.2; echo $?
   1
```

23.8 ups flavor

The **ups flavor** command with no options returns the flavor of the machine. If a flavor level is specified (e.g., **-0**, **-1** ...), it returns the flavor according to that level. **ups flavor** generates a flavor table if the **-H** option is used. The flavor levels and the term *flavor table* are defined in section 23.8.4 *More Detailed Description*.

23.8.1 Command Syntax

```
% ups flavor [<options>]
```

23.8.2 Commonly Used Options

See section 23.8.3 *All Valid Options* for descriptions of each option.

-f <flavor> Or one of **-0**, **-1**, **-2**, **-3**, **-4**

-H <flavor> Alone or together with one of **-0**, **-1**, **-2**, **-3**, **-4**

-l

Table 23.8.2-a:

-f <flavor>	Or one of **-0**, **-1**, **-2**, **-3**, **-4**
-H <flavor>	Alone or together with one of **-0**, **-1**, **-2**, **-3**, **-4**
-l	

23.8.3 All Valid Options

-? (**"-?"** for **csh**) Prints command description and option usage information to screen.

-f <flavor> Specifies flavor.

-H <flavor> Specifies host flavor family from which to build a flavor table. If used with any of **-0**, **-1**, **-2**, **-3**, **-4**, specifies corresponding level of specified host flavor family.

-l Produces a flavor table.

-v(**vvv**) Prints out extra debugging information.

-Z Times the command.

-0 Specifies flavor as NULL; equivalent to **-f NULL**

-1 Specifies flavor as OS value up to the generic OS; e.g., on a SunOS machine it is equivalent to **-f SunOS**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX.

-2 Specifies flavor for product instance on local and distribution nodes as OS value up to the version; e.g., equivalent to **-f SunOS+5**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.

-3 Specifies flavor for product instance on local and distribution nodes upto the release of the version; e.g., equivalent to **-f SunOS+5.6**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.2.

-4 Specifies flavor for product instance on local and distribution nodes upto the patch number of the release; e.g., equivalent to **-f SunOS+5.6.1**; if given together with **-H IRIX+6.2.2**, flavor is then specified as IRIX+6.2.2.

Table 23.8.3-a:

-? ("-?" for csh)	Prints command description and option usage information to screen.
-f <flavor>	Specifies flavor.
-H <flavor>	Specifies host flavor family from which to build a flavor table. If used with any of **-0**, **-1**, **-2**, **-3**, specifies corresponding level of specified host flavor family.
-l	Produces a flavor table.
-v(vvv)	Prints out extra debugging information.
-Z	Times the command.
-0	Specifies flavor as NULL; equivalent to **-f NULL**
-1	Specifies flavor as OS value up to the generic OS; e.g., on a SunOS machine it is equivalent to **-f SunOS**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX.
-2	Specifies flavor for product instance on local and distribution nodes as OS value up to the version; e.g., equivalent to **-f SunOS+5**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.

Table 23.8.3-a:

-3	Specifies flavor for product instance on local and distribution nodes upto the release of the version; e.g., equivalent to **-f SunOS+5.6**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.2.
-4	Specifies flavor for product instance on local and distribution nodes upto the patch number of the release; e.g., equivalent to **-f SunOS+5.6.1**; if given together with **-H IRIX+6.2.2**, flavor is then specified as IRIX+6.2.2.

23.8.4 More Detailed Description

The **ups flavor** command returns flavor information about the machine issuing the command, or for a flavor requested via the **-H** option. When entered with no options, the command returns the full OS specification of the machine.

When entered with the **-l** (long) option, **ups flavor** returns what we call a *flavor table*, which is a list including every level of specificity for a flavor that you could use to find or declare a product instance. For example, on a SunOS machine (release 5.6) it outputs:

```
% ups flavor -l
   SunOS+5.6
   SunOS+5
   SunOS
   NULL
   ANY
```

If **-H <flavor>** is used with **-l**, **ups flavor** builds a flavor table for the flavor given by **-H**. This is useful if you're not sure what levels are allowable for a particular basic flavor. The flavor table lists flavors starting at the level you specify. Compare the following two commands and output:

```
% ups flavor -lH IRIX+6.6
   IRIX+6.6
   IRIX+6
   IRIX
   NULL
   ANY
% ups flavor -lH IRIX
   IRIX
   NULL
   ANY
```

You can specify a particular level using the number options: **-0**, **-1**, **-2**, **-3**, **-4**, of which **-3** is the most highly specified; for example:

```
% ups flavor -4
```

```
   SunOS+5.6-2
% ups flavor -3
   SunOS+5.6
% ups flavor -2
   SunOS+5
% ups flavor -1
   SunOS
% ups flavor -0
   NULL
```

23.8.5 ups flavor Examples

Find full flavor specification of machine

```
% ups flavor
```

This command returns the full OS specification of the machine upto the build number for the patch (when these levels of specification exist), for example:

```
SunOS+5.6.1-4
```

Create a flavor table for machine's OS

```
% ups flavor -l
```

This command returns a flavor table for the flavor of the machine. For example, on a (fictional) SunOS+5.6 machine it outputs:

```
SunOS+5.6.1-4
SunOS+5.6.1
SunOS+5.6
SunOS+5
SunOS
NULL
ANY
```

Find flavor specification of machine, at different levels

```
% ups flavor -4
```

The **-4** option requests the machine's flavor as the most significant OS specification or the full specification, e.g.,:

```
SunOS+5.6.1
% ups flavor -3
```

The **-3** option requests the machine's flavor as the most significant OS specification or the full specification, e.g.,:

```
SunOS+5.6
```

```
% ups flavor -1
```

The **-1** option requests the machine's flavor as the OS value up to the generic OS, e.g.,:

```
SunOS
```

```
% ups flavor -0
```

This always returns the NULL string.

Create a flavor table for host flavor, at different levels

```
% ups flavor -lH IRIX+6.6
```

This creates a flavor table listing flavors starting at the level specified via **-H**, in this case "level 3":

```
IRIX+6.6
IRIX+6
IRIX
NULL
ANY
```

```
% ups flavor -lH IRIX+6
```

This creates a flavor table listing flavors starting at the level specified via **-H**, in this case "level 2":

```
IRIX+6
IRIX
NULL
ANY
```

23.9 ups get

The **ups get** command is rarely used by anyone except product developers/maintainers. Currently it can only be used with the **-F** option. **ups get -F** lists any files on the local node which were distributed with the specified product instance(s) and which are maintained outside of the product root directory. The list does not include table files, for which the location is maintained in the version file.

The **ups get** command was designed primarily for use by **UPD**, which calls it internally. As such it is rarely used outside of that context. In a future release, **ups get** may acquire additional functions.

23.9.1 Command Syntax

```
% ups get -F [<other_options>] <product> [<version>]
```

23.9.2 All valid options

-? (**"-?"** for **csh**) Prints command description and option usage information to screen

-c Finds product instance chained to “current”

-d Finds product instance chained to “development”

-f <flavor> Finds product instance of specified flavor. If specified and no exact match is found, the command fails. Multiple values accepted, but **UPS** looks only at first in list.

-F Prints to screen a list of files that are associated with the product but which are maintained external to the products area (excluding table file)

-g <chainName> Finds product instance chained to **<chainName>**

-H <flavor> Specifies flavor and builds a flavor list for that family starting at the level specified. **UPS** finds the best match instance for the specified flavor family. Multiple values accepted, but **UPS** looks only at first in list.

-m <tableFileName> Specifies table file name

-M <tableFileDir> Specifies table file directory

-n Finds product instance chained to “new”

-o Finds product instance chained to “old”

-q <qualifierList> Finds product instance with the specified qualifiers (required and/or optional)

-r <prodRootDir> Specifies the product root directory

-t Finds product instance chained to “test”

-U <upsDir> Specifies location of `ups` directory; default value is **ups**

-v(**vvv**) Prints out extra debugging information.

-V Does not delete the temporary script files or partially installed products when command finishes; instead lists them on the screen

-z <databaseList> Specifies the database(s) in which to look for the product and its dependencies

-Z Times the command

Table 23.9.2-a:

-? ("-?" for csh)	Prints command description and option usage information to screen
-c	Finds product instance chained to “current”
-d	Finds product instance chained to “development”
-f <flavor>	Finds product instance of specified flavor. If specified and no exact match is found, the command fails. Multiple values accepted, but **UPS** looks only at first in list.
-F	Prints to screen a list of files that are associated with the product but which are maintained external to the products area (excluding table file)
-g <chainName>	Finds product instance chained to **<chainName>**
-H <flavor>	Specifies flavor and builds a flavor list for that family starting at the level specified. **UPS** finds the best match instance for the specified flavor family. Multiple values accepted, but **UPS** looks only at first in list.
-m <tableFileName>	Specifies table file name
-M <tableFileDir>	Specifies table file directory
-n	Finds product instance chained to “new”
-o	Finds product instance chained to “old”
-q <qualifierList>	Finds product instance with the specified qualifiers (required and/or optional)
-r <prodRootDir>	Specifies the product root directory
-t	Finds product instance chained to “test”
-U <upsDir>	Specifies location of **ups** directory; default value is **ups**
-v(vvv)	Prints out extra debugging information.
-V	Does not delete the temporary script files or partially installed products when command finishes; instead lists them on the screen
-z <databaseList>	Specifies the database(s) in which to look for the product and its dependencies

Table 23.9.2-a:

-Z	Times the command

23.9.3 ups get Example

```
% ups get -F ups
```

In the database on the machine used, the **UPS** product has a few files maintained outside of the product root directory. This command returns the output:

```
/fnal/ups/db/.upsfiles/configure/v4_4a_OSF1+V4_/current
/fnal/ups/db/.upsfiles/configure/v4_4a_OSF1+V4_/uncurrent
/fnal/ups/db/.upsfiles/configure/v4_4a_OSF1+V4_/unconfigure
```

23.10 ups help

The **ups help** command lists all the **UPS** commands with brief definitions. There are no options for this command.

23.10.1 ups help Example

```
% ups help
   UPS commands:

   for specific command options use "ups COMMAND -?"

              configure   : Environmentally configure a product
   instance.
             copy       : Allow one instance of a product to be
   declared "like" another.
             declare    : Add a product instance or a chain to
   a UPS Database.
              depend      : List (for a specified UPS product
   instance) UPS product
                              requirements or all UPS product
   instances that have the
                             specified UPS product instance as a
   requirement.
             exist      : Determine if a setup command with the
   same options
                            would likely succeed.
```

```
          flavor      : Print flavor of a machine, optionally
by level, or table
                        generated for searching
           get         : Return a list of all files that are
needed  by a product
                         instance and do not live under the
product root directory.
           help        : Output help information for all UPS
commands
           list        : List UPS Database information about
product instances.
          modify      : Allow editor modification of the UPS
Database files.
                       The altered files are verified before
being rewritten.
           setup       : Prepare the environment in order to
be able to use a product
                        instance.
           start       : Perform any necessary actions for a
product instance at system
                        boot.
           stop        : Perform any necessary actions for a
product instance at
                        system shutdown.
         tailor     : Perform any product instance tailoring
that needs to be done
                            once (specify hardware device
location) or needs user input.
           touch       : Will change a ups file modify time
(MODIFIED) to current time
                           (it will probaly also change the
modifier (MODIFIER)).
            unconfigure : Undo any actions performed by the
configure command.
           undeclare   : Remove a product instance from a UPS
Database.
                         if chain(s) are specified ONLY the
chain(s) will version will
                        be removed
         unsetup     : Return the environment to a pre-setup
state.
           verify       : Check the specified instances for
correct formatting and
                        information.
```

23.11 ups list

The **ups list** command returns information about the declared product instances in a **UPS** database. Two output styles are provided: a formatted one that is easy for users to read, and a condensed one for parsing by a subsequent command or a script.

23.11.1 Command Syntax

```
% ups list [<options>] [<product>] [<version>]
```

23.11.2 Commonly Used Options

See section 23.11.3 *All Valid Options* for descriptions of each option.

-a

-f <flavorList> Or one of **-0**, **-1**, **-2**, **-3**, **-4**, or **-H** (alone or together with one of **-0**, **-1**, **-2**, **-3**, **-4**)

-g <chainName> Or one of **-c**, **-d**, **-n**, **-o**, **-t**

-K <keywordList>

-l

-q <qualifierList>

-z <databaseList>

Table 23.11.2-a:

-a	
-f <flavorList>	Or one of **-0**, **-1**, **-2**, **-3**, **-4**, or **-H** (alone or together with one of **-0**, **-1**, **-2**, **-3**, **-4**)
-g <chainName>	Or one of **-c**, **-d**, **-n**, **-o**, **-t**
-K <keywordList>	
-l	
-q <qualifierList>	
-z <databaseList>	

23.11.3 All Valid Options

-? (**"-?"** for **csh**) Prints command description and option usage information to screen

-a Lists all instances that match the other options given on command line

-c Finds product instance(s) chained to “current”

-d Finds product instance(s) chained to “development”

-f <flavorList> Described below under “The flavor options”.

-g <chainName> Finds product instance(s) chained to **<chainName>**

-H <flavorList> Described below under “The flavor options”.

-K <keywordList> Returns values of specified keywords only; valid keywords are listed in section 28.4 *List of Supported Keywords*

-l Produces a long listing; lists all keywords for the product instance(s)

-m <tableFileName> Specifies table file name

-M <tableFileDir> Specifies table file directory

-n Finds product instance(s) chained to “new”

-o Finds product instance(s) chained to “old”

-q <qualifierList> Finds product instance(s) with the specified qualifiers (required and/or optional)

-r <prodRootDir> Specifies the product root directory

-t Finds product instance(s) chained to “test”

-U <upsDir> Specifies location of `ups` directory; default value is **`ups`**

-v(**vvv**) Prints out extra debugging information.

-V Does not delete the temporary script files or partially installed products when command finishes; instead lists them on the screen

-z <databaseList> Specifies the database(s) in which to look for the product instance(s)

-Z Times the command (does not include time for sourcing of temp file for **`setup/unsetup`**)

Table 23.11.3-a:

-? ("-?" for csh)	Prints command description and option usage information to screen
-a	Lists all instances that match the other options given on command line
-c	Finds product instance(s) chained to “current”
-d	Finds product instance(s) chained to “development”
-f <flavorList>	Described below under “The flavor options”.
-g <chainName>	Finds product instance(s) chained to **<chainName>**
-H <flavorList>	Described below under “The flavor options”.
-K <keywordList>	Returns values of specified keywords only; valid keywords are listed in section 28.4 *List of Supported Keywords*
-l	Produces a long listing; lists all keywords for the product instance(s)
-m <tableFileName>	Specifies table file name
-M <tableFileDir>	Specifies table file directory
-n	Finds product instance(s) chained to “new”
-o	Finds product instance(s) chained to “old”
-q <qualifierList>	Finds product instance(s) with the specified qualifiers (required and/or optional)
-r <prodRootDir>	Specifies the product root directory
-t	Finds product instance(s) chained to “test”
-U <upsDir>	Specifies location of **ups** directory; default value is **ups**
-v(vvv)	Prints out extra debugging information.
-V	Does not delete the temporary script files or partially installed products when command finishes; instead lists them on the screen
-z <databaseList>	Specifies the database(s) in which to look for the product instance(s)

Table 23.11.3-a:

-Z	Times the command (does not include time for sourcing of temp file for **setup/unsetup**)

The flavor options

Flavor may be specified using **-f**, using **-H** by itself or in combination with any of **-0**, **-1**, **-2**, **-3**, **-4**, or just using one of **-0**, **-1**, **-2**, **-3**, **-4**. These options are not valid with each other (except **-H** with a number option).

-f <flavorList> Finds product instance(s) of specified flavor. If specified and no exact match is found, the command fails. If multiple values specified, **UPS** looks for instances matching all the values.

-H <flavorList> Specifies flavor and builds a flavor list for that family starting at the level specified.

Can be used alone (without an accompanying number option). **UPS** looks for instances matching all the flavor levels. If multiple values specified (usually of different flavor families), **UPS** looks for instances matching all the flavor levels of each value.

If used with any of **-0**, **-1**, **-2**, **-3**, **-4**, **UPS** looks for product instances of specified level of that flavor; e.g., **-2H IRIX+6.2** is equivalent to **-f IRIX+6**. If multiple values specified (usually of different flavor families), **UPS** looks for instances matching each value, according to the accompanying number option.

Note: if **-a** and **-H** are both used, and **-H** is used without a number option, then **-H** has no effect; all flavors get listed.

-0 Specifies flavor as NULL; equivalent to **-f NULL**

-1 Specifies flavor as OS value up to the generic OS; e.g., on a SunOS machine it is equivalent to **-f SunOS**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX.

-2 Specifies flavor for product instance on local and distribution nodes as OS value up to the version; e.g., equivalent to **-f SunOS+5**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.

-3 Specifies flavor for product instance on local and distribution nodes upto the release of the version; e.g., equivalent to **-f SunOS+5.6**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.2.

-4 Specifies flavor for product instance on local and distribution nodes upto the patch number of the release; e.g., equivalent to **-f SunOS+5.6.2**; if given together with **-H IRIX+6.2.1**, flavor is then specified as IRIX+6.2.1.

Table 23.11.3-b:

-f <flavorList>	Finds product instance(s) of specified flavor. If specified and no exact match is found, the command fails. If multiple values specified, **UPS** looks for instances matching all the values.
-H <flavorList>	Specifies flavor and builds a flavor list for that family starting at the level specified. Can be used alone (without an accompanying number option). **UPS** looks for instances matching all the flavor levels. If multiple values specified (usually of different flavor families), **UPS** looks for instances matching all the flavor levels of each value. Can be used alone (without an accompanying number option). **UPS** looks for instances matching all the flavor levels. If multiple values specified (usually of different flavor families), **UPS** looks for instances matching all the flavor levels of each value.
	If used with any of **-0**, **-1**, **-2**, **-3**, **-4**, **UPS** looks for product instances of specified level of that flavor; e.g., **-2H IRIX+6.2** is equivalent to **-f IRIX+6**. If multiple values specified (usually of different flavor families), **UPS** looks for instances matching each value, according to the accompanying number option.
	Note: if **-a** and **-H** are both used, and **-H** is used without a number option, then **-H** has no effect; all flavors get listed.
-0	Specifies flavor as NULL; equivalent to **-f NULL**
-1	Specifies flavor as OS value up to the generic OS; e.g., on a SunOS machine it is equivalent to **-f SunOS**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX.
-2	Specifies flavor for product instance on local and distribution nodes as OS value up to the version; e.g., equivalent to **-f SunOS+5**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.
-3	Specifies flavor for product instance on local and distribution nodes upto the release of the version; e.g., equivalent to **-f SunOS+5.6**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.2.
-4	Specifies flavor for product instance on local and distribution nodes upto the patch number of the release; e.g., equivalent to **-f SunOS+5.6.2**; if given together with **-H IRIX+6.2.1**, flavor is then specified as IRIX+6.2.1.

23.11.4 More Detailed Description

`ups list` is useful for finding out what products are in the database that you use, what the current version of a product is for your machine's flavor, and other information. Product installers and other administrative users can use it to get detailed information about a product's installation and to find product files.

You can specify the information you want contained in the output by including various options in the command. As is standard in **UPS**, if no chain, version or flavor is specified, and **`-a`** (for all instances) is *not* specified, **UPS** returns only the instance declared as current for the best-matched flavor of the requesting machine.

Formatted Output Style

One output style is for visual parsing (this is the default output, which we call *formatted*), with output that looks like:

```
% ups list xemacs

   DATABASE=/usr/upsII/ups_database/declared/oss
           Product=xemacs  Version=v19_14  Flavor=SunOS+5
                   Qualifiers=""   Chain=current
```

Notice that the product name, version, flavor, qualifiers and chain(s) are the default fields that get returned. The database (first line of the output) is included as a header, not as part of the per-instance data. Each piece of data returned for an instance is preceded by its keyword for identification.

You cannot choose arbitrary output fields for the selected instances using this output format. However, you can use the **`-l`** option to give an exhaustive listing of information contained in the **UPS** database about the requested product.

If $PRODUCTS contains multiple databases, output is returned for each one and labelled accordingly, for example:

```
% ups list xemacs

   DATABASE=/usr/upsII/ups_database/devel
   DATABASE=/usr/upsII/ups_database/declared/oss
           Product=xemacs  Version=v19_14  Flavor=SunOS+5
                   Qualifiers=""   Chain=current

   DATABASE=/usr/upsII/ups_database/declared/afs
           Product=xemacs  Version=v19_14  Flavor=SunOS+5
                   Qualifiers=""   Chain=current
```

Notice that the first database is listed in the output even though it doesn't contain any instances of the product that match the request.

Condensed Output Style

The other output format is a script-readable (or *condensed*) format, provided to allow parsing by a subsequent command. Use the **-K** option to request output in the condensed format. The **-K** option requires an argument list specifying which fields to include in the output, for example:

```
% ups list -K product:version:flavor xemacs
   "xemacs" "v19_14" "SunOS+5"
```

The plus sign (**+**) argument, e.g., **-K+**, is a shorthand for requesting the default fields **product:version:flavor:qualifiers:chain**, for example:

```
% ups list -K+ xemacs
   "xemacs" "v19_14" "SunOS+5" "" "current"
```

Some common keyword arguments used with the **-K** option are:

PRODUCT	product name
FLAVOR	product instance flavor
VERSION	product version
QUALIFIERS	additional instance specification information often used to indicate compilation options used by developer
CHAIN	product instance chain
+	all of the above
DATABASE (or DB)	the **UPS** database path; useful if more than one on system
DECLARER	logon id of person who declared the instance
DECLARED	date/time that product instance was declared
MODIFIER	logon id of person who modified/updated the instance
MODIFIED	date/time that product instance was modified/updated

Table 23.11.4-a:

PRODUCT	product name
FLAVOR	product instance flavor
VERSION	product version
QUALIFIERS	additional instance specification information often used to indicate compilation options used by developer
CHAIN	product instance chain
+	all of the above

Table 23.11.4-a:

DATABASE (or DB)	the **UPS** database path; useful if more than one on system
DECLARER	logon id of person who declared the instance
DECLARED	date/time that product instance was declared
MODIFIER	logon id of person who modified/updated the instance
MODIFIED	date/time that product instance was modified/updated

If $PRODUCTS contains multiple databases, output is returned for selected products in all of them. However, the database is identified for each output line only if the keyword DATABASE or DB is included in the argument string (e.g., **`-K+:DB`** requests the standard output fields followed by the database path).

A few of the keywords allow the "at" symbol, @ to be prepended, which provides a sort of shorthand for long path names:

@PROD_DIR	entire path for the directory where the product is installed (usually equivalent to PROD_DIR_PREFIX/PROD_DIR)
@TABLE_FILE	entire path for the table file
@UPS_DIR	product's `ups` directory; if it is not an absolute path, then it is taken relative to @PROD_DIR

Table 23.11.4-b:

@PROD_DIR	entire path for the directory where the product is installed (usually equivalent to PROD_DIR_PREFIX/PROD_DIR)
@TABLE_FILE	entire path for the table file
@UPS_DIR	product's `ups` directory; if it is not an absolute path, then it is taken relative to @PROD_DIR

The full list of keywords that can be used with **`ups list -K`** and **`upd list -K`** follows, with descriptions:

Keyword	Description
ARCHIVE_FILE	archive file name/location; useful with **`upd list`**
AUTHORIZED_NODES	authorized nodes; "all nodes" represented by an asterisk (*) in output
CATMAN_SOURCE_DIR	location of catman files (formatted man page files) included with instance

Keyword	Description
CATMAN_TARGET_DIR	directory into which catman files are to be copied
CHAIN	chain name
COMPILE_DIR	directory in which the compile file resides
COMPILE_FILE	the name of the file containing compiled functions (see Chapter 38: *Use of Compile Scripts in Table Files*)
@COMPILE_FILE	entire path to the file containing compiled functions
DECLARED	the date/time that the instance was declared to **UPS** or declared with a chain Note: often has multiple values, one for each declaration (e.g., for subsequent chain declarations)
DECLARER	userid of user that performed the declaration Note: often has multiple values, one for each declaration (e.g., for subsequent chain declarations)
DESCRIPTION	product description
FLAVOR	product instance flavor
HTML_SOURCE_DIR	location of html files included with instance *not supported in UPS v4*
HTML_TARGET_DIR	directory into which html files are to be copied *not supported in UPS v4*
INFO_SOURCE_DIR	location of Info files included with instance
INFO_TARGET_DIR	directory into which Info files are to be copied
MAN_SOURCE_DIR	location of unformatted man page files included with instance
MAN_TARGET_DIR	directory into which formatted man pages are to be copied
MODIFIED	last time the associated instance was changed Note: often has multiple values, one for each declaration/modification (e.g., for subsequent chain declarations)
MODIFIER	userid of user that modified the instance Note: often has multiple values, one for each declaration/modification (e.g., for subsequent chain declarations)
NEWS_SOURCE_DIR	location of news files included with instance *not supported in UPS v4*
NEWS_TARGET_DIR	directory into which news files are to be copied (for posting to a newsgroup) *not supported in UPS v4*
ORIGIN	master source file; see option **-D** in Chapter 25: *Generic Command Option Descriptions*
PRODUCT	product name

Keyword	Description
PROD_DIR	product root directory (usually defined relative to PROD_DIR_PREFIX, below)
@PROD_DIR	entire path to product root directory
PROD_DIR_PREFIX	product root directory prefix (area where all or most product instances are maintained)
QUALIFIERS	additional instance specification information often used to indicate compilation options used by developer
SETUPS_DIR	location of `setups.[c]sh` files and other **UPS** initialization files
STATISTICS	flag to record statistics for specified products
TABLE_DIR	location of table file
TABLE_FILE	name of table file
@TABLE_FILE	entire path for the table file.
UPD_USERCODE_DIR	Directory where **UPD** configuration files are maintained
UPS_DIR	location of `ups` directory (if not absolute path, then taken relative to PROD_DIR, usually)
@UPS_DIR	entire path to `ups` directory
VERSION	product version

23.11.5 ups list Examples

List all current products

```
% ups list
```

The simplest way to request a listing of all the current products installed in your default **UPS** database is to use the **`ups list`** command with no options or arguments. The per-product output spans a few lines, though, and can be cumbersome, e.g.,:

```
DATABASE=/afs/fnal.gov/ups/db
       Product=admintools     Version=v1     Flavor=SunOS+5
               Qualifiers=""   Chain=current

        Product=afsemu  Version=v1_2    Flavor=NULL
               Qualifiers=""   Chain=current

...

        Product=xpdf    Version=v0_7    Flavor=SunOS+5
               Qualifiers=""   Chain=current

        Product=zephyr  Version=v2_0_4  Flavor=SunOS+5
               Qualifiers=""   Chain=current
```

Use of the **`-K`** option with the **`+`** argument, e.g.,

```
% ups list -K+
```

provides the same information as the previous example, but condensed to one line per product:

```
"admintools" "v1" "SunOS+5" "" "current"
"afsemu" "v1_2" "NULL" "" "current"
...
"xpdf" "v0_7" "SunOS+5" "" "current"
"zephyr" "v2_0_4" "SunOS+5" "" "current"
```

Instead of using the **`+`** argument to get the default fields, you can specify particular fields:

```
% ups list -K product:version
```

This command outputs a list of product names and version numbers for all the current products installed in your default **UPS** database, e.g.,:

```
"admintools" "v1"
"afsemu" "v1_2"
...
"xpdf" "v0_7"
"zephyr" "v2_0_4"
```

List standard information for default instance of product

```
% ups list emacs
```

This command requests the standard output fields for the default instance of **emacs**, using the formatted output:

```
DATABASE=/afs/fnal.gov/ups/db
        Product=emacs   Version=v19_34b Flavor=SunOS+5
                Qualifiers=""   Chain=current
```

Addition of the **-K+** construction, e.g.,

```
% ups list -K+ emacs
```

requests the same information as the previous example, but in condensed format:

```
"emacs" "v19_34b" "SunOS+5" "" "current"
```

Using **-a** for all, e.g.,

List standard information for all instances of product

```
% ups list -a emacs
```

requests the standard output fields for all instances of **emacs**, using the formatted output:

```
DATABASE=/afs/fnal.gov/ups/db
        Product=emacs   Version=v19_30a Flavor=AIX+3
                Qualifiers=""   Chain=""

        Product=emacs   Version=v19_30a  Flavor=IRIX+5
                Qualifiers=""   Chain=""

        Product=emacs   Version=v19_30a Flavor=SunOS+5
                Qualifiers=""   Chain=""
...

        Product=emacs   Version=v19_34b Flavor=SunOS+5
                Qualifiers=""   Chain=current
```

Use of the **-K** option with the **+** argument, e.g.,

```
% ups list -aK+ emacs
```

requests the same information as the previous example, but in condensed format (**-K+**):

```
"emacs" "v19_30a" "AIX+3" "" ""
"emacs" "v19_30a" "IRIX+5" "" ""
"emacs" "v19_30a" "SunOS+5" "" ""
...
"emacs" "v19_34b" "SunOS+5" "" "current"
```

List standard information for all instances of product for your machine's flavor

```
% ups list -a2K+ emacs
```

To request information on all instances of a product limited to the OS of your machine, you can include the **-f** option (for flavor), or enter a command like this one, where **-a** is for all, **-2** is for the “level 2” designation of your machine’s OS, and **-K+** is for condensed output:

```
"emacs" "v19_30a" "SunOS+5" "" ""
"emacs" "v19_34" "SunOS+5" "" ""
"emacs" "v19_34b" "SunOS+5" "" "current"
```

List specific keywords for a product

If your installation has multiple databases defined in $PRODUCTS, it is useful to include the keyword for the database (**DB**) in the **-K** argument list, e.g.,

```
% ups list -aK+:DB emacs
```

This example is similar to the previous one, but the database path is included at the end:

```
"emacs" "v19_30a" "AIX+3" "" "" "/afs/fnal.gov/ups/db"
"emacs" "v19_30a" "IRIX+5" "" "" "/afs/fnal.gov/ups/db"
"emacs"        "v19_30a"        "AIX+3"        ""        ""
"/usr/products/ups_database/main"
"emacs"        "v19_30a"        "IRIX+5"        ""        ""
"/usr/products/ups_database/main"
...
```

```
% ups list -K product:version admintools
```

This specifies particular fields to be output with the **-K** option for the default instance of the product **admintools**, producing:

```
"admintools" "v1"
```

List all keywords for a product (long listing)

```
% ups list emacs -l
```

This command requests detailed information (**-l** for long listing) about the default instance of the product **emacs**. Administrative users may often need this level of detailed information about a product. The **-l** option is not valid with the **-K** option. The output looks like this:

```
DATABASE=/afs/fnal.gov/ups/db
        Product=emacs   Version=v19_34b Flavor=SunOS+5
                Qualifiers=""   Chain=current
                Declared="1997-07-15 00.00.00 GMT:1997-03-21
00.00.00 GMT"
                Declarer="root:colossus"
                Modified=":1997-03-21 00.00.00 GMT"
                Modifier=":colossus"

Home=/afs/fnal.gov/products/SunOS+5/emacs/v19_34b
                No Compile Directive
                Authorized, Nodes=*
                UPS_Dir="ups"
                Table_Dir=""
                Table_File="v19_34b.table"
                Archive_File=""
                Description=""
                Action=setup
```

```
                        proddir()
                        setupenv()

sourceRequired(${UPS_UPS_DIR}/setup.${UPS_SHELL},UPS_ENV)
                Action=Configure
                        ProdDir()

execute(${UPS_UPS_DIR}/configure,UPS_ENV)
                        UnProdDir()
                Action=unConfigure
                        ProdDir()

execute(${UPS_UPS_DIR}/unconfigure,UPS_ENV)
                        UnProdDir()
                Action=Current
                        ProdDir()

execute(${UPS_UPS_DIR}/configure,UPS_ENV)

execute(${UPS_UPS_DIR}/current,UPS_ENV)
                        UnProdDir()
```

```
                    Action=unCurrent
                           ProdDir()

   execute(${UPS_UPS_DIR}/uncurrent,UPS_ENV)
                           UnProdDir()
```

This gives a fairly long list. It is often more convenient to use the **-K** option with a list of keywords for the specific fields you need.

Use "ups list -K" to locate product root directory, table file and ups directory

ups list -K can be used to locate a product's root directory, table file, and ups directory when used with the keywords corresponding to these quantities

Compare the following three commands and their output. **UPS_DIR** represents the location of the product's ups directory. If it is not an absolute path, then it is taken relative to **@PROD_DIR**, if specified (as shown in the second command). **@UPS_DIR** is the absolute path.

```
% ups list -K @PROD_DIR teledata
   "/afs/fnal.gov/ups/teledata/v1_0/NULL"
% ups list -KUPS_DIR teledata
   "ups"
% ups list -K@UPS_DIR teledata
   "/afs/fnal.gov/ups/db/teledata/v1_0/NULL/ups"
```

Compare the following two commands and their output. **table_file** represents only the name of the table file, not its path. **@table_file** is the entire path for the table file. See section 29.4 *Determination of ups Directory and Table File Locations* for information on how **UPS** determines the table file directory.

```
% ups list -Ktable_file teledata
   "v1_0.table"
% ups list -K@table_file teledata
   "/afs/fnal.gov/ups/db/teledata/v1_0.table"
```

Parse output from "ups list -K" in perl

Here we provide guidance on parsing **ups list** output in **perl**, with all the appropriate quoting and spacing. The following example first defines the file handle UPS_LIST_OUTPUT to contain the piped output of the command **$UPS_DIR/bin/ups list -aK+** (where **$UPS_DIR** is translated

inside of **perl** via the **`$ENV{UPS_DIR}`** statement, which is the translation of the environmental variable *UPS_DIR*). It then defines the array `@fields`, and parses the output into a set of five variables.

```
open(UPS_LIST_OUTPUT, "| $ENV{UPS_DIR}/bin/ups list -aK+");
while (<UPS_LIST_OUTPUT> ) {
# break into the array @fields:
@fields = m/\"((?:[^\"\\]|\\.)*)\"/g;
# then do things with $field[0] $field[1] ...
# (in this case,
# $field[0] = product name
# $field[1] = version
# $field[2] = flavor
# $field[3] = qualifiers (colon-separated list)
# $field[4] = chains (colon-separated list)
}
```

Say the output from your **`ups list -K`** command looks like this:

```
"ups" "v4_4" "IRIX+5" "" "current"
```

then the `@fields` array contains the variables:

```
$field[0] = ups
$field[1] = v4_4
$field[2] = IRIX+5
$field[3] =
$field[4] = current
```

Alternatively, you could parse the output this way:

```
($product, $version, $flavor, $qualifiers, $chains) =
m/\"((?:[^\"\\]|\\.)*)\"/g;
```

and then you'd have:

```
$product = ups
$version = v4_4
$flavor = IRIX+5
$qualifiers =
$chains = current
```

Parsing output from "ups list -K" in a sh script

You can parse the output from a command of the form **`ups list -K`** by piping it into a "*while* loop" **sh** script. Here is an example; explanations follow the code:

```
ups list -K... |
while read line
do
     eval set : $line
     shift
# now do things with $1 $2 $3...
```

```
done
```

As the condition of the *while* loop, the **`read line`** command reads the lines of output into the variable *line*. To get rid of the quotes, the loop runs **`eval set : $line`** on each line (this syntax ensures that the **`set`** command actually sets the variables `$1`, `$2`, and so on, instead of setting shell behavior in case the first argument starts with a dash). The **`shift`** that follows then gets rid of the colon.

23.12 ups modify

The **`ups modify`** command allows you to manually edit any of the database product files. It performs syntax and content validation before and after the editing session.

23.12.1 Command Syntax

```
% ups modify [<options>] <product> [<version>]
```

23.12.2 Commonly Used Options

See section 23.12.3 *All Valid Options* for descriptions of each option.

`-E <editor>`

`-f <flavor>` Or one of **`-0`**, **`-1`**, **`-2`**, **`-3`**, **`-4`**, or **`-H`** (alone or together with one of **`-0`**, **`-1`**, **`-2`**, **`-3`**, **`-4`**)

`-g <chainName>` Or one of **`-c`**, **`-d`**, **`-n`**, **`-o`**, **`-t`**

`-N <fileName>`

`-q <qualifierList>`

`-z <databaseList>`

Table 23.12.2-a:

`-E <editor>`	
`-f <flavor>`	Or one of **`-0`**, **`-1`**, **`-2`**, **`-3`**, **`-4`**, or **`-H`** (alone or together with one of **`-0`**, **`-1`**, **`-2`**, **`-3`**, **`-4`**)
`-g <chainName>`	Or one of **`-c`**, **`-d`**, **`-n`**, **`-o`**, **`-t`**

Table 23.12.2-a:

-N <file-Name>	
-q <qual-ifier-List>	
-z <data-baseList>	

23.12.3 All Valid Options

-? (**"-?"** for **csh**) Prints command description and option usage information to screen

-a Operates on all instances that match the other options given on command line

-E <editor> Invokes the specified editor. If not given, uses the editor specified by $EDITOR. If not set, uses **vi** by default.

-f <flavor> Finds product instance of specified flavor. If specified and no exact match is found, the command fails. Multiple values accepted, but **UPS** looks only at first in list.

-H <flavor> Specifies flavor and builds a flavor list for that family starting at the level specified. **UPS** finds the best match instance for the specified flavor family. Multiple values accepted, but **UPS** looks only at first in list.

-m <tableFileName> Specifies table file name

-M <tableFileDir> Specifies table file directory

-N <fileName> Specifies file to be checked and edited

-p "<description>" Specifies product description

-q <qualifierList> Finds product instance with the specified qualifiers (required and/or optional)

-r <prodRootDir> Specifies the product root directory

-T <path or URL> Specifies archive file path or URL. This is used only for declarations in distribution databases for which products are maintained in tar or gzip (archived) format.

-U <upsDir> Specifies location of `ups` directory; default value is **ups**

-v(**vvv**) Prints out extra debugging information.

-V Does not delete the temporary script files or partially installed products when command finishes; instead lists them on the screen

-z <databaseList> Specifies the database(s) in which to look for the product

-Z Times the command

Table 23.12.3-a:

-? ("-?" for csh)	Prints command description and option usage information to screen
-a	Operates on all instances that match the other options given on command line
-E <editor>	Invokes the specified editor. If not given, uses the editor specified by $EDITOR. If not set, uses **vi** by default.
-f <flavor>	Finds product instance of specified flavor. If specified and no exact match is found, the command fails. Multiple values accepted, but **UPS** looks only at first in list.
-H <flavor>	Specifies flavor and builds a flavor list for that family starting at the level specified. **UPS** finds the best match instance for the specified flavor family. Multiple values accepted, but **UPS** looks only at first in list.
-m <tableFileName>	Specifies table file name
-M <tableFileDir>	Specifies table file directory
-N <fileName>	Specifies file to be checked and edited
-p "<description>"	Specifies product description
-q <qualifierList>	Finds product instance with the specified qualifiers (required and/or optional)
-r <prodRootDir>	Specifies the product root directory
-T <path or URL>	Specifies archive file path or URL. This is used only for declarations in distribution databases for which products are maintained in tar or gzip (archived) format.
-U <upsDir>	Specifies location of `ups` directory; default value is **`ups`**
-v(vvv)	Prints out extra debugging information.

Table 23.12.3-a:

-V	Does not delete the temporary script files or partially installed products when command finishes; instead lists them on the screen
-z <dataBaseList>	Specifies the database(s) in which to look for the product
-Z	Times the command

23.12.4 More Detailed Description

ups modify performs the following steps (if you specify the file using **-N**, the menu will not appear):

- presents menu of files that you can edit and asks you to either select one or quit
- verifies pre-modification contents of file (runs **ups verify**)
- starts up the editor given by **-E <editor>** or, if that is not specified, then $EDITOR, if set. If neither is specified, it starts up **vi** by default.
- makes a copy of the file to be edited
- pulls copy of file into the editor
- after user exits the editor, runs **ups verify** on the edited file
- if the validation succeeds, writes the new file over the old one and quits
- if the validation does not succeed, provides informational messages, and quits
- if no changes made to file, again presents menu of files

Internal Processes

- Bring up requested file in specified editor
- Verify the pre-edited file
- Verify the edited file before overwriting
- Process MODIFY action
- Execute the temp file

23.12.5 ups modify Example

```
% ups modify teledata v1_0 -N $MYDB/teledata/v1_0.version
```

In this example, we select the version file (via **-N**) for the product **teledata** v1_0 (default flavor, no qualifiers). Since **-E** is not given, **UPS** will use the editor set in $EDITOR, or **vi** if that variable is not set. First, **UPS** runs **ups verify** and produces the output:

```
Pre modification verification pass complete.
```

No errors were detected. The version file is next displayed in the editor.

1) To illustrate an unsuccessful validation, we add a bogus line:

```
TESTKEYWORD = value
```

and save and quit. **UPS** returns the following messages, and we opt to save the erroneous change:

```
INFORMATIONAL: Unexpected key word 'TESTKEYWORD' in
'/home/t1/aheavey/upsII/decl
ared/teledata/v1_0.version', line 17
INFORMATIONAL: Unexpected key word 'TESTKEYWORD' in
'/home/t1/aheavey/upsII/decl
ared/teledata/v1_0.version', line 17
Post modification verification pass complete.
Do you wish to save this modification [y/n] ? y
```

UPS quits, saving the file as we requested.

2) To illustrate successful validation, we'll correct the error introduced above. We run the same **ups modify** command. **UPS** finds the error during the pre-edit validation:

```
INFORMATIONAL: Unexpected key word 'TESTKEYWORD' in
'/home/t1/aheavey/upsII/decl
ared/teledata/v1_0.version', line 17
INFORMATIONAL: Unexpected key word 'TESTKEYWORD' in
'/home/t1/aheavey/upsII/decl
ared/teledata/v1_0.version', line 17
Pre modification verification pass complete.
```

We remove the incorrect line from the version file, then save and quit. **UPS** displays the following message, and we elect to save the change (**y**):

```
Post modification verification pass complete.
Do you wish to save this modification [y/n] ? y
```

UPS quits, saving the file as requested.

23.13 ups parent

The **ups parent** command can be used to determine which products depend on the specified product instance(s) as declared in the (local) database. This command is useful when deciding whether a product instance can be removed without causing problems for other products. However, it is very slow (see section 23.13.4 *More Detailed Description*). If you need to look things up frequently, run the command with the **-a** option, dump the output to a file, and search in there.

23.13.1 Command Syntax

```
% ups parent [<options>] <product> [<version>]
```

23.13.2 Commonly Used Options

See section 23.11.3 *All Valid Options* for descriptions of each option.

-a

-f <flavorList> Or one of **-0**, **-1**, **-2**, **-3**, **-4**, or **-H** (alone or together with one of **-0**, **-1**, **-2**, **-3**, **-4**)

-g <chainName> Or one of **-c**, **-d**, **-n**, **-o**, **-t**

-K <keywordList>

-l

-q <qualifierList>

-z <databaseList>

Table 23.13.2-a:

-a	
-f <flavorList>	Or one of **-0**, **-1**, **-2**, **-3**, **-4**, or **-H** (alone or together with one of **-0**, **-1**, **-2**, **-3**, **-4**)
-g <chainName>	Or one of **-c**, **-d**, **-n**, **-o**, **-t**
-K <keywordList>	
-l	

Table 23.13.2-a:

-q <qual-ifier-List>	
-z <data-baseList>	

23.13.3 All Valid Options

-? (**"-?"** for **csh**) Prints command description and option usage information to screen

-a Lists all instances that match the other options given on command line

-c Finds product instance(s) chained to “current”

-d Finds product instance(s) chained to “development”

-f <flavorList> Described below under “The flavor options”.

-g <chainName> Finds product instance(s) chained to **<chainName>**

-H <flavorList> Described below under “The flavor options”.

-K <keywordList> Returns values of specified keywords only; valid keywords are listed in section 28.4 *List of Supported Keywords*

-l Produces a long listing; lists all keywords for the product instance(s)

-m <tableFileName> Specifies table file name

-M <tableFileDir> Specifies table file directory

-n Finds product instance(s) chained to “new”

-o Finds product instance(s) chained to “old”

-q <qualifierList> Finds product instance(s) with the specified qualifiers (required and/or optional)

-r <prodRootDir> Specifies the product root directory

-t Finds product instance(s) chained to “test”

-U <upsDir> Specifies location of `ups` directory; default value is **ups**

-v(**vvv**) Prints out extra debugging information.

-V Does not delete the temporary script files or partially installed products when command finishes; instead lists them on the screen

-z <databaseList> Specifies the database(s) in which to look for the product instance(s)

-Z Times the command (does not include time for sourcing of temp file for **setup/unsetup**)

Table 23.13.3-a:

-? ("-?" for csh)	Prints command description and option usage information to screen
-a	Lists all instances that match the other options given on command line
-c	Finds product instance(s) chained to “current”
-d	Finds product instance(s) chained to “development”
-f <flavorList>	Described below under “The flavor options”.
-g <chainName>	Finds product instance(s) chained to **<chainName>**
-H <flavorList>	Described below under “The flavor options”.
-K <keywordList>	Returns values of specified keywords only; valid keywords are listed in section 28.4 *List of Supported Keywords*
-l	Produces a long listing; lists all keywords for the product instance(s)
-m <tableFileName>	Specifies table file name
-M <tableFileDir>	Specifies table file directory
-n	Finds product instance(s) chained to “new”
-o	Finds product instance(s) chained to “old”
-q <qualifierList>	Finds product instance(s) with the specified qualifiers (required and/or optional)
-r <prodRootDir>	Specifies the product root directory
-t	Finds product instance(s) chained to “test”
-U <upsDir>	Specifies location of `ups` directory; default value is **ups**
-v(vvv)	Prints out extra debugging information.

Table 23.13.3-a:

-V	Does not delete the temporary script files or partially installed products when command finishes; instead lists them on the screen
-z <dataBaseList>	Specifies the database(s) in which to look for the product instance(s)
-Z	Times the command (does not include time for sourcing of temp file for **setup/unsetup**)

The flavor options

Flavor may be specified using **-f**, using **-H** by itself or in combination with any of **-0**, **-1**, **-2**, **-3**, **-4**, or just using one of **-0**, **-1**, **-2**, **-3**, **-4**. These options are not valid with each other (except **-H** with a number option).

-f <flavorList> Finds product instance(s) of specified flavor. If specified and no exact match is found, the command fails. If multiple values specified, **UPS** looks for instances matching all the values.

-H <flavorList> Specifies flavor and builds a flavor list for that family starting at the level specified.

Can be used alone (without an accompanying number option). **UPS** looks for instances matching all the flavor levels. If multiple values specified (usually of different flavor families), **UPS** looks for instances matching all the flavor levels of each value.

If used with any of **-0**, **-1**, **-2**, **-3**, **-4**, **UPS** looks for product instances of specified level of that flavor; e.g., **-2H IRIX+6.2** is equivalent to **-f IRIX+6**. If multiple values specified (usually of different flavor families), **UPS** looks for instances matching each value, according to the accompanying number option.

Note: if **-a** and **-H** are both used, and **-H** is used without a number option, then **-H** has no effect; all flavors get listed.

-0 Specifies flavor as NULL; equivalent to **-f NULL**

-1 Specifies flavor as OS value up to the generic OS; e.g., on a SunOS machine it is equivalent to **-f SunOS**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX.

-2 Specifies flavor for product instance on local and distribution nodes as OS value up to the version; e.g., equivalent to **-f SunOS+5**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.

-3 Specifies flavor for product instance on local and distribution nodes upto the release of the version; e.g., equivalent to **-f SunOS+5.6**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.2.

-4 Specifies flavor for product instance on local and distribution nodes upto the patch number of the release; e.g., equivalent to **-f SunOS+5.6.2**; if given together with **-H IRIX+6.2.1**, flavor is then specified as IRIX+6.2.1.

Table 23.13.3-b:

-f <flavorList>	Finds product instance(s) of specified flavor. If specified and no exact match is found, the command fails. If multiple values specified, **UPS** looks for instances matching all the values.
-H <flavorList>	Specifies flavor and builds a flavor list for that family starting at the level specified. Can be used alone (without an accompanying number option). **UPS** looks for instances matching all the flavor levels. If multiple values specified (usually of different flavor families), **UPS** looks for instances matching all the flavor levels of each value. Can be used alone (without an accompanying number option). **UPS** looks for instances matching all the flavor levels. If multiple values specified (usually of different flavor families), **UPS** looks for instances matching all the flavor levels of each value.
	If used with any of **-0**, **-1**, **-2**, **-3**, **-4**, **UPS** looks for product instances of specified level of that flavor; e.g., **-2H IRIX+6.2** is equivalent to **-f IRIX+6**. If multiple values specified (usually of different flavor families), **UPS** looks for instances matching each value, according to the accompanying number option.
	Note: if **-a** and **-H** are both used, and **-H** is used without a number option, then **-H** has no effect; all flavors get listed.
-0	Specifies flavor as NULL; equivalent to **-f NULL**
-1	Specifies flavor as OS value up to the generic OS; e.g., on a SunOS machine it is equivalent to **-f SunOS**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX.
-2	Specifies flavor for product instance on local and distribution nodes as OS value up to the version; e.g., equivalent to **-f SunOS+5**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.
-3	Specifies flavor for product instance on local and distribution nodes upto the release of the version; e.g., equivalent to **-f SunOS+5.6**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.2.
-4	Specifies flavor for product instance on local and distribution nodes upto the patch number of the release; e.g., equivalent to **-f SunOS+5.6.2**; if given together with **-H IRIX+6.2.1**, flavor is then specified as IRIX+6.2.1.

23.13.4 More Detailed Description

The **`ups parent`** command is REALLY slow; its companion command **`upd parent`** takes about an hour, and that's running it from a machine on-site! The **`ups parent`** command runs:

```
% ups depend product version -f flavor -q qualifiers -H
  hostflavor
```

for every `product version -f flavor -q qualfiers` you have, and for every `hostflavor` mentioned in the database. So, for example, in the Fermilab AFS database, that's 4198 product instances times 32 flavors. You do the math!

It takes about the same time to run the command on multiple products, e.g.,:

```
% ups parent -a
```

or

```
% ups parent product1, product2
```

as to run the command on a single product, e.g.,:

```
% ups parent product1
```

so for multiple products, use comma-separated option sets rather than multiple **`ups parent`** commands. If you look things up frequently, we recommend that you run:

```
% ups parent -a
```

dump the output to a file, and search in the file for the information you need.

In a future release we may teach **UPS** to cache parent info.

23.13.5 ups parent Example

If you want to know what products depend on **tcl**, run:

```
% ups parent tcl
```

and you get something like the following (the percentage counts up):

```
100% completed.
tcl v8_0_2 -f Linux+2 -z /afs/fnal.gov/ups/db
|__tk v8_0_2 -f Linux+2 -z /afs/fnal.gov/ups/db
   |__blt v2_3 -f Linux+2 -z /afs/fnal.gov/ups/db
   |__expect v5_25 -f Linux+2 -z /afs/fnal.gov/ups/db
   |  |__buildmanager devel -f NULL -z /fnal/ups/db
   |  |__buildmanager v1_10 -f NULL -z /fnal/ups/db
   |  |__buildmanager v1_11 -f NULL -z /fnal/ups/db
   |  |__buildmanager v1_3 -f NULL -z /afs/fnal.gov/ups/db
   |  |__buildmanager v1_5 -f NULL -z /afs/fnal.gov/ups/db
```

```
   |  |__buildmanager v1_6 -f NULL -z /afs/fnal.gov/ups/db
   |  |__buildmanager v1_7 -f NULL -z /afs/fnal.gov/ups/db
   |  |__buildmanager v1_8 -f NULL -z /afs/fnal.gov/ups/db
   |  |__buildmanager v1_9 -f NULL -z /afs/fnal.gov/ups/db
   |  |__exmh v2_0_2 -f NULL -z /afs/fnal.gov/ups/db
   |__ical v2_2 -f Linux+2 -z /afs/fnal.gov/ups/db
   |  |__tktools v8_0_2 -f NULL -z /afs/fnal.gov/ups/db
   |__python v1_5_2 -f Linux+2 -z /afs/fnal.gov/ups/db
   |  |__ngop b0_5 -f Linux -z /afs/fnal.gov/ups/db
         |     |__ngop   v1_1   -f   Linux   -q   monitor   -z
/afs/fnal.gov/ups/db
   |  |__ngop v1_1 -f Linux -z /afs/fnal.gov/ups/db
         |     |__ngop   v1_2   -f   Linux   -q   monitor   -z
/afs/fnal.gov/ups/db
   |  |__ngop v1_2 -f Linux -z /afs/fnal.gov/ups/db
   |  |__pydb v1_01 -f NULL -z /afs/fnal.gov/ups/db
   |  |__python_msql v2_0b -f Linux+2 -z /afs/fnal.gov/ups/db
   |__rolodex v1_1 -f NULL -z /afs/fnal.gov/ups/db
   |  |__(tktools v8_0_2 -f NULL -z /afs/fnal.gov/ups/db )
   |__rolodex v1_2 -f NULL -z /fnal/ups/db
   |__tclx v8_0_2 -f Linux+2 -z /afs/fnal.gov/ups/db
   |__tkman v1_7_5 -f Linux+2 -z /afs/fnal.gov/ups/db
   |  |__(tktools v8_0_2 -f NULL -z /afs/fnal.gov/ups/db )
   |__tksession v1_2 -f NULL -z /afs/fnal.gov/ups/db
   |  |__(tktools v8_0_2 -f NULL -z /afs/fnal.gov/ups/db )
   |__(tktools v8_0_2 -f NULL -z /afs/fnal.gov/ups/db )
   |__ucm v5_1 -f NULL -z /afs/fnal.gov/ups/db
   |__ucm v5_2 -f NULL -z /afs/fnal.gov/ups/db
   |__ucm vdev -f NULL -z /afs/fnal.gov/ups/db
```

In the AFS database this takes about four minutes on a fast machine.

23.14 ups start

The **`ups start`** command is used to invoke appropriately configured products at system boot time. This is used primarily for products that need to load drivers, start daemons or perform other actions required at boot time. This command is generally not run manually, rather it gets executed from within a **UPS** control file.

The **`ups stop`** command (see section 23.15 *ups stop*) is used to stop products that are started this way.

23.14.1 Command Syntax

```
% ups start [<options>] <product> [<version>]
```

23.14.2 Commonly Used Options

See section 23.14.3 *All Valid Options* for descriptions of each option.

-f <flavor> Or one of **-0**, **-1**, **-2**, **-3**, **-4**, or **-H** (alone or together with one of **-0**, **-1**, **-2**, **-3**, **-4**)

-g <chainName> Or one of **-c**, **-d**, **-n**, **-o**, **-t**

-q <qualifierList>

-v(**vvv**)

-w

-z <databaseList>

Table 23.14.2-a:

-f <flavor>	Or one of **-0**, **-1**, **-2**, **-3**, **-4**, or **-H** (alone or together with one of **-0**, **-1**, **-2**, **-3**, **-4**)
-g <chainName>	Or one of **-c**, **-d**, **-n**, **-o**, **-t**
-q <qualifierList>	
-v(vvv)	
-w	
-z <databaseList>	

23.14.3 All Valid Options

-? (**"-?"** for **csh**) Prints command description and option usage information to screen

-c Finds product instance chained to “current”

-d Finds product instance chained to “development”

-f <flavor> Described below under “The flavor options”.

-g <chainName> Finds product instance chained to **<chainName>**

-H <flavor> Described below under “The flavor options”.

-m <tableFileName>Specifies table file name

-M <tableFileDir>Specifies table file directory

-n Finds product instance chained to “new”

-o Finds product instance chained to “old”

-O "<flags>" Sets the value of $UPS_OPTIONS to **<flags>**.

-P Requires **UPS** to rely only on information supplied on the command line to locate the product instance (prevents **UPS** from searching in a database)

-q <qualifierList>Finds product instance with the specified qualifiers (required and/or optional)

-r <prodRootDir>Specifies the product root directory

-s Lists what command would do; but does not execute the command

-t Finds product instance chained to “test”

-U <upsDir> Specifies location of `ups` directory; default value is **ups**

-v(**vvv**) Prints out extra debugging information.

-V Does not delete the temporary script files or partially installed products when command finishes; instead lists them on the screen

-w Stops the product first, then restarts it

-z <databaseList>Specifies the database(s) in which to look for the product and its dependencies

-Z Times the command

Table 23.14.3-a:

-? ("-?" for csh)	Prints command description and option usage information to screen
-c	Finds product instance chained to “current”
-d	Finds product instance chained to “development”
-f <flavor>	Described below under “The flavor options”.
-g <chainName>	Finds product instance chained to **<chainName>**

Table 23.14.3-a:

-H <flavor>	Described below under “The flavor options”.
-m <tableFileName>	Specifies table file name
-M <tableFileDir>	Specifies table file directory
-n	Finds product instance chained to “new”
-o	Finds product instance chained to “old”
-O "<flags>"	Sets the value of $UPS_OPTIONS to **<flags>**.
-P	Requires **UPS** to rely only on information supplied on the command line to locate the product instance (prevents **UPS** from searching in a database)
-q <qualifierList>	Finds product instance with the specified qualifiers (required and/or optional)
-r <prodRootDir>	Specifies the product root directory
-s	Lists what command would do; but does not execute the command
-t	Finds product instance chained to “test”
-U <upsDir>	Specifies location of **ups** directory; default value is **ups**
-v(vvv)	Prints out extra debugging information.
-V	Does not delete the temporary script files or partially installed products when command finishes; instead lists them on the screen
-w	Stops the product first, then restarts it
-z <databaseList>	Specifies the database(s) in which to look for the product and its dependencies
-Z	Times the command

The flavor options

Flavor may be specified using **-f**, using **-H** by itself or in combination with any of **-0**, **-1**, **-2**, **-3**, **-4**, or just using one of **-0**, **-1**, **-2**, **-3**, **-4**. These options are not valid with each other (except **-H** with a number option).

-f <flavor> Finds product instance of specified flavor. If specified and no exact match is found, the command fails. Multiple values accepted, but **UPS** looks only at first in list.

-H <flavor> Specifies flavor and builds a flavor list for that family starting at the level specified. Multiple values accepted, but **UPS** looks only at first in list.

Can be used alone (without an accompanying number option). In this case, **UPS** finds the best match instance for the specified flavor family.

If used with any of **-0**, **-1**, **-2**, **-3**, **-4**, **UPS** finds the product instance of specified level of that flavor; e.g., **-2H IRIX+6.2** is equivalent to **-f IRIX+6**.

-0 Specifies flavor as NULL; equivalent to **-f NULL**

-1 Specifies flavor as OS value up to the generic OS; e.g., on a SunOS machine it is equivalent to **-f SunOS**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX.

-2 Specifies flavor for product instance on local and distribution nodes as OS value up to the version; e.g., equivalent to **-f SunOS+5**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.

-3 Specifies flavor for product instance on local and distribution nodes upto the release of the version; e.g., equivalent to **-f SunOS+5.6**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.2.

-4 Specifies flavor for product instance on local and distribution nodes up to the patch number of the release; e.g., equivalent to **-f SunOS+5.6.2**; if given together with **-H IRIX+6.2.1**, flavor is then specified as IRIX+6.2.1.

Table 23.14.3-b:

-f <flavor>	Finds product instance of specified flavor. If specified and no exact match is found, the command fails. Multiple values accepted, but **UPS** looks only at first in list.
-H <flavor>	Specifies flavor and builds a flavor list for that family starting at the level specified. Multiple values accepted, but **UPS** looks only at first in list. Can be used alone (without an accompanying number option). In this case, **UPS** finds the best match instance for the specified flavor family. If used with any of **-0**, **-1**, **-2**, **-3**, **-4**, **UPS** finds the product instance of specified level of that flavor; e.g., **-2H IRIX+6.2** is equivalent to **-f IRIX+6**.

Table 23.14.3-b:

-0	Specifies flavor as NULL; equivalent to **-f NULL**
-1	Specifies flavor as OS value up to the generic OS; e.g., on a SunOS machine it is equivalent to **-f SunOS**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX.
-2	Specifies flavor for product instance on local and distribution nodes as OS value up to the version; e.g., equivalent to **-f SunOS+5**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.
-3	Specifies flavor for product instance on local and distribution nodes upto the release of the version; e.g., equivalent to **-f SunOS+5.6**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.2.
-4	Specifies flavor for product instance on local and distribution nodes upto the patch number of the release; e.g., equivalent to **-f SunOS+5.6.2**; if given together with **-H IRIX+6.2.1**, flavor is then specified as IRIX+6.2.1.

23.14.4 More Detailed Description

Please see Chapter 15: *Automatic UPS Product Startup and Shutdown* for a detailed description of the use of the **ups start** command. In brief, it covers:

- Configuring your machine to allow automatic startup/shutdown
- Installing a **UPS** product to start and/or stop automatically, for which you need to:
  - Determine if auto start/stop feature is enabled
  - Determine if product is appropriate for autostart
  - Edit control file(s)
  - Restart the system

Internal Processes

- Check node authorization
- If necessary, process STOP action
- Process START action
- Execute the temp file

23.14.5 ups start Examples

Run the command interactively

This command first stops (**-w**) the default instance of the product **apache**, and then restarts it:

```
% ups start -w apache
```

It prints to screen messages of the format:

```
Stopping Apache server for fnkits.fnal.gov on port 8000
Stopping Apache server for fnkits.fnal.gov on port 8000
account updadmin
```

Run the command from a control file

This command starts the default instance of the product **ObjectCenter** for the flavor SunOS, and requests verbose output (**-v**).

```
% ups start -v -f SunOS ObjectCenter
```

If the logon id is *root*, the line in the control file may look like:

```
ups start -v -f SunOS ObjectCenter
```

If the logon id is other than *root*, the line in the control file must look like:

```
/bin/su - products -c "ups start -v -f SunOS ObjectCenter"
```

23.15 ups stop

The **ups stop** command is used to stop appropriately configured products at system shutdown. This command is generally not run manually, rather it gets executed from within a **UPS** control file, and operates on products that were invoked using **ups start** (see section 23.14 *ups start*). Typically, these products are ones which need to load drivers, start daemons or perform other actions required at boot time.

23.15.1 Command Syntax

```
% ups stop [<options>] <product> [<version>]
```

23.15.2 Commonly Used Options

See section 23.15.3 *All Valid Options* for descriptions of each option.

-f <flavor> Or one of **-0**, **-1**, **-2**, **-3**, **-4**, or **-H** (alone or together with one of **-0**, **-1**, **-2**, **-3**, **-4**)

-g <chainName> Or one of **-c**, **-d**, **-n**, **-o**, **-t**

-q <qualifierList>

-v(**vvv**)

-z <databaseList>

Table 23.15.2-a:

-f <flavor>	Or one of **-0**, **-1**, **-2**, **-3**, **-4**, or **-H** (alone or together with one of **-0**, **-1**, **-2**, **-3**, **-4**)
-g <chainName>	Or one of **-c**, **-d**, **-n**, **-o**, **-t**
-q <qualifierList>	
-v(vvv)	
-z <databaseList>	

23.15.3 All Valid Options

-? (**"-?"** for **csh**) Prints command description and option usage information to screen

-c Finds product instance chained to “current”

-d Finds product instance chained to “development”

-f <flavor> Described below under “The flavor options”.

-g <chainName> Finds product instance chained to **<chainName>**

-H <flavor> Described below under “The flavor options”.

-m <tableFileName> Specifies table file name

-M <tableFileDir> Specifies table file directory

-n Finds product instance chained to “new”

-o Finds product instance chained to “old”

-O "<flags>" Sets the value of $UPS_OPTIONS to **<flags>**.

-P Requires **UPS** to rely only on information supplied on the command line to locate the product instance (prevents **UPS** from searching in a database)

-q <qualifierList> Finds product instance with the specified qualifiers (required and/or optional)

-r <prodRootDir> Specifies the product root directory

-s Lists what command would do; but does not execute the command

-t Finds product instance chained to “test”

-U <upsDir> Specifies location of `ups` directory; default value is **ups**

-v(**vvv**) Prints out extra debugging information.

-V Does not delete the temporary script files or partially installed products when command finishes; instead lists them on the screen

-z <databaseList> Specifies the database(s) in which to look for the product and its dependencies

-Z Times the command

Table 23.15.3-a:

-? ("-?" for csh)	Prints command description and option usage information to screen
-c	Finds product instance chained to “current”
-d	Finds product instance chained to “development”
-f <flavor>	Described below under “The flavor options”.
-g <chainName>	Finds product instance chained to **<chainName>**
-H <flavor>	Described below under “The flavor options”.
-m <tableFileName>	Specifies table file name
-M <tableFileDir>	Specifies table file directory
-n	Finds product instance chained to “new”
-o	Finds product instance chained to “old”
-O "<flags>"	Sets the value of $UPS_OPTIONS to **<flags>**.

Table 23.15.3-a:

-P	Requires **UPS** to rely only on information supplied on the command line to locate the product instance (prevents **UPS** from searching in a database)
-q <qualifier-List>	Finds product instance with the specified qualifiers (required and/or optional)
-r <prodRootDir>	Specifies the product root directory
-s	Lists what command would do; but does not execute the command
-t	Finds product instance chained to "test"
-U <upsDir>	Specifies location of `ups` directory; default value is **ups**
-v(vvv)	Prints out extra debugging information.
-V	Does not delete the temporary script files or partially installed products when command finishes; instead lists them on the screen
-z <databaseList>	Specifies the database(s) in which to look for the product and its dependencies
-Z	Times the command

The flavor options

Flavor may be specified using **-f**, using **-H** by itself or in combination with any of **-0**, **-1**, **-2**, **-3**, **-4**, or just using one of **-0**, **-1**, **-2**, **-3**, **-4**. These options are not valid with each other (except **-H** with a number option).

-f <flavor> Finds product instance of specified flavor. If specified and no exact match is found, the command fails. Multiple values accepted, but **UPS** looks only at first in list.

-H <flavor> Specifies flavor and builds a flavor list for that family starting at the level specified. Multiple values accepted, but **UPS** looks only at first in list.

Can be used alone (without an accompanying number option). In this case, **UPS** finds the best match instance for the specified flavor family.

If used with any of **-0**, **-1**, **-2**, **-3**, **-4**, **UPS** finds the product instance of specified level of that flavor; e.g., **-2H IRIX+6.2** is equivalent to **-f IRIX+6**.

-0 Specifies flavor as NULL; equivalent to **-f NULL**

-1 Specifies flavor as OS value up to the generic OS; e.g., on a SunOS machine it is equivalent to **-f SunOS**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX.

-2 Specifies flavor for product instance on local and distribution nodes as OS value up to the version; e.g., equivalent to **-f SunOS+5**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.

-3 Specifies flavor for product instance on local and distribution nodes up to the release of the version; e.g., equivalent to **-f SunOS+5.6**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.2.

-4 Specifies flavor for product instance on local and distribution nodes upto the patch number of the release; e.g., equivalent to **-f SunOS+5.6.2**; if given together with **-H IRIX+6.2.1**, flavor is then specified as IRIX+6.2.1.

Table 23.15.3-b:

-f <flavor>	Finds product instance of specified flavor. If specified and no exact match is found, the command fails. Multiple values accepted, but **UPS** looks only at first in list.
-H <flavor>	Specifies flavor and builds a flavor list for that family starting at the level specified. Multiple values accepted, but **UPS** looks only at first in list. Can be used alone (without an accompanying number option). In this case, **UPS** finds the best match instance for the specified flavor family. If used with any of **-0**, **-1**, **-2**, **-3**, **UPS** finds the product instance of specified level of that flavor; e.g., **-2H IRIX+6.2** is equivalent to **-f IRIX+6**.
-0	Specifies flavor as NULL; equivalent to **-f NULL**
-1	Specifies flavor as OS value up to the generic OS; e.g., on a SunOS machine it is equivalent to **-f SunOS**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX.
-2	Specifies flavor for product instance on local and distribution nodes as OS value up to the version; e.g., equivalent to **-f SunOS+5**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.
-3	Specifies flavor for product instance on local and distribution nodes upto the release of the version; e.g., equivalent to **-f SunOS+5.6**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.2.
-4	Specifies flavor for product instance on local and distribution nodes upto the patch number of the release; e.g., equivalent to **-f SunOS+5.6.2**; if given together with **-H IRIX+6.2.1**, flavor is then specified as IRIX+6.2.1.

23.15.4 More Detailed Description

Please see Chapter 15: *Automatic UPS Product Startup and Shutdown* for a detailed description of the use of the **`ups stop`** command. In brief, it covers:

- Configuring your machine to allow automatic startup/shutdown
- Installing a **UPS** product to start and/or stop automatically, for which you need to:
  - · Determine if auto start/stop feature is enabled
  - · Determine if product is appropriate for autostart
  - · Edit control file(s)
  - · Restart the system

Internal Processes

- Check node authorization
- Process STOP action
- Execute the temp file

23.15.5 ups stop Examples

Run the command interactively

This command stops the default instance of the product **apache**:

```
% ups stop apache
```

It prints to screen a message of the format:

```
Stopping Apache server for fnkits.fnal.gov on port 8000
```

Run the command from a control file

This command stops the default instance of the product **ObjectCenter** for the flavor SunOS, and requests verbose output (**`-v`**).

```
% ups stop -v -f SunOS ObjectCenter
```

If the logon id is *root*, the line in the control file may look like:

```
ups stop -v -f SunOS ObjectCenter
```

If the logon id is other than *root*, the line in the control file must look like:

```
/bin/su - products -c "ups stop -v -f SunOS ObjectCenter"
```

23.16 ups tailor

For any product instance whose table file includes a TAILOR action, the **ups tailor** command must be run manually at product installation time after the product is declared to the database in order to execute this action. A TAILOR action typically includes installation functions which require input from the installer.

23.16.1 Command Syntax

```
% ups tailor [<options>] <product> [<version>]
```

23.16.2 Commonly Used Options

See section 23.16.3 *All Valid Options* for descriptions of each option.

-f <flavor> Or one of **-0**, **-1**, **-2**, **-3**, **-4**, or **-H** (alone or together with one of **-0**, **-1**, **-2**, **-3**, **-4**)

-g <chainName> Or one of **-c**, **-d**, **-n**, **-o**, **-t**

-q <qualifierList>

-v(**vvv**)

-z <databaseList>

Table 23.16.2-a:

-f <flavor>	Or one of **-0**, **-1**, **-2**, **-3**, **-4**, or **-H** (alone or together with one of **-0**, **-1**, **-2**, **-3**, **-4**)
-g <chainName>	Or one of **-c**, **-d**, **-n**, **-o**, **-t**
-q <qualifierList>	
-v(vvv)	
-z <databaseList>	

23.16.3 All Valid Options

`-?` (**`"-?"`** for **csh**) Prints command description and option usage information to screen

`-c` Finds product instance chained to “current”

`-d` Finds product instance chained to “development”

`-f <flavor>` Described below under “The flavor options”.

`-g <chainName>` Finds product instance chained to **`<chainName>`**

`-H <flavor>` Described below under “The flavor options”.

`-K <keywordList>` Returns values of specified keywords only; valid keywords are listed in section 28.4 *List of Supported Keywords*

`-m <tableFileName>` Specifies table file name

`-M <tableFileDir>` Specifies table file directory

`-n` Finds product instance chained to “new”

`-o` Finds product instance chained to “old”

`-O "<flags>"` Sets the value of $UPS_OPTIONS to **`<flags>`**.

`-P` Requires **UPS** to rely only on information supplied on the command line to locate the product instance (prevents **UPS** from searching in a database)

`-q <qualifierList>` Finds product instance with the specified qualifiers (required and/or optional)

`-r <prodRootDir>` Specifies the product root directory

`-s` Lists what command would do; but does not execute the command

`-t` Finds product instance chained to “test”

`-U <upsDir>` Specifies location of `ups` directory; default value is **`ups`**

`-v`(**`vvv`**) Prints out extra debugging information.

`-V` Does not delete the temporary script files or partially installed products when command finishes; instead lists them on the screen

`-z <databaseList>` Specifies the database(s) in which to look for the product and its dependencies

`-Z` Times the command (does not include time for sourcing of temp file for **`setup/unsetup`**)

Table 23.16.3-a:

-? ("-?" for csh)	Prints command description and option usage information to screen
-c	Finds product instance chained to “current”
-d	Finds product instance chained to “development”
-f <flavor>	Described below under “The flavor options”.
-g <chainName>	Finds product instance chained to **<chainName>**
-H <flavor>	Described below under “The flavor options”.
-K <keywordList>	Returns values of specified keywords only; valid keywords are listed in section 28.4 *List of Supported Keywords*
-m <tableFileName>	Specifies table file name
-M <tableFileDir>	Specifies table file directory
-n	Finds product instance chained to “new”
-o	Finds product instance chained to “old”
-O "<flags>"	Sets the value of $UPS_OPTIONS to **<flags>**.
-P	Requires **UPS** to rely only on information supplied on the command line to locate the product instance (prevents **UPS** from searching in a database)
-q <qualifierList>	Finds product instance with the specified qualifiers (required and/or optional)
-r <prodRootDir>	Specifies the product root directory
-s	Lists what command would do; but does not execute the command
-t	Finds product instance chained to “test”
-U <upsDir>	Specifies location of `ups` directory; default value is **ups**
-v(vvv)	Prints out extra debugging information.
-V	Does not delete the temporary script files or partially installed products when command finishes; instead lists them on the screen

Table 23.16.3-a:

-z <dataBaseList>	Specifies the database(s) in which to look for the product and its dependencies
-Z	Times the command (does not include time for sourcing of temp file for **setup/unsetup**)

The flavor options

Flavor may be specified using **-f**, using **-H** by itself or in combination with any of **-0**, **-1**, **-2**, **-3**, **-4**, or just using one of **-0**, **-1**, **-2**, **-3**, **-4**. These options are not valid with each other (except **-H** with a number option).

-f <flavor> Finds product instance of specified flavor. If specified and no exact match is found, the command fails. Multiple values accepted, but **UPS** looks only at first in list.

-H <flavor> Specifies flavor and builds a flavor list for that family starting at the level specified. Multiple values accepted, but **UPS** looks only at first in list.

Can be used alone (without an accompanying number option). In this case, **UPS** finds the best match instance for the specified flavor family.

If used with any of **-0**, **-1**, **-2**, **-3**, **-4**, **UPS** finds the product instance of specified level of that flavor; e.g., **-2H IRIX+6.2** is equivalent to **-f IRIX+6**.

-0 Specifies flavor as NULL; equivalent to **-f NULL**

-1 Specifies flavor as OS value up to the generic OS; e.g., on a SunOS machine it is equivalent to **-f SunOS**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX.

-2 Specifies flavor for product instance on local and distribution nodes as OS value up to the version; e.g., equivalent to **-f SunOS+5**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.

-3 Specifies flavor for product instance on local and distribution nodes upto the release of the version; e.g., equivalent to **-f SunOS+5.6**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.2.

-4	Specifies flavor for product instance on local and distribution nodes upto the patch number of the release; e.g., equivalent to **-f SunOS+5.6.2**; if given together with **-H IRIX+6.2.1**, flavor is then specified as IRIX+6.2.1.

Table 23.16.3-b:

-f <flavor>	Finds product instance of specified flavor. If specified and no exact match is found, the command fails. Multiple values accepted, but **UPS** looks only at first in list.
-H <flavor>	Specifies flavor and builds a flavor list for that family starting at the level specified. Multiple values accepted, but **UPS** looks only at first in list. Can be used alone (without an accompanying number option). In this case, **UPS** finds the best match instance for the specified flavor family. If used with any of **-0**, **-1**, **-2**, **-3**, **-4**, **UPS** finds the product instance of specified level of that flavor; e.g., **-2H IRIX+6.2** is equivalent to **-f IRIX+6**.
-0	Specifies flavor as NULL; equivalent to **-f NULL**
-1	Specifies flavor as OS value up to the generic OS; e.g., on a SunOS machine it is equivalent to **-f SunOS**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX.
-2	Specifies flavor for product instance on local and distribution nodes as OS value up to the version; e.g., equivalent to **-f SunOS+5**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.
-3	Specifies flavor for product instance on local and distribution nodes upto the release of the version; e.g., equivalent to **-f SunOS+5.6**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.2.
-4	Specifies flavor for product instance on local and distribution nodes upto the patch number of the release; e.g., equivalent to **-f SunOS+5.6.2**; if given together with **-H IRIX+6.2.1**, flavor is then specified as IRIX+6.2.1.

23.16.4 More Detailed Description

Tailoring is the aspect of the product implementation that requires input from the product installer (e.g., specifying the location of hardware devices for a software driver package). If the product requires tailoring, a file is usually supplied in the format of an interactive executable (script or compiled binary), and it is run via a function under the ACTION=TAILOR line in the table file. Configure, tailor and other supplemental installation scripts are usually maintained in the product's `$<PRODUCT>_DIR/ups` directory. You must explicitly tailor the product instance using the **UPS** command **ups tailor**; tailoring is *not* performed automatically.

Tailoring is generally allowed on any node of a cluster, however we strongly recommend that you perform any node-specific tailoring from that node, or flavor-specific tailoring from a node of that flavor to avoid mismatches.

Internal Processes

- Check node authorization
- Process TAILOR action
- Execute the temp file

23.16.5 ups tailor Example

As an example, we show the tailor process for the product **apache**:

```
% ups tailor apache
   Current configuration nicknames are:
   kits    kits8k
   You can:
       a)dd a new server configuration
       q)uit the tailor script
   Which would you like? [aq]? a
   Webserver      alias/name      (e.g.      www-xx.fnal.gov)?
   www-demo.fnal.gov
   Webserver nickname [demo]?
   Webserver port number [80]?
   Webserver effective user-id [wwwsrv]?
   Webserver effective group-id [www]?
   Webserver admin id [wwwadm]?
   Mail    address    for    admin    stuff    [mengel@fnal.gov]?
   demo-admin@fnal.gov
   Directory for CGI executables [/fnal/www/demo/cgi-bin]?
   Directory /fnal/www/demo/cgi-bin doesn't exist, make it [y]?
   y
   Root of served files [/fnal/www/demo/html]?
   Directory /fnal/www/demo/html doesn't exist, make it [y]? y
   Raw Log file directory [/var/adm/www/demo]? /tmp/demo
   Log file Summary directory [/fnal/www/demo/html/logs]?
   Directory  /fnal/www/demo/html/logs  doesn't  exist,  make  it
   [y]? y
   Current configuration nicknames are:
   demo    kits    kits8k
```

```
You can:
    a)dd a new server configuration
    q)uit the tailor script
Which would you like? [aq]? q
```

23.17 ups touch

The **ups touch** command changes the values of the keywords MODIFIED and MODIFIER in the version file to the current time and the current user, respectively.

If you make any changes to a product's database files by hand (e.g., not via **ups modify**), you should run **ups touch** afterwards.[1] You can also run it to prevent an update if you choose to run **upd update** on several product instances at a time.

23.17.1 Command Syntax

```
% ups touch [<options>] <product> [<version>]
```

23.17.2 Commonly Used Options

See section 23.17.3 *All Valid Options* for descriptions of each option.

-f <flavor> Or one of **-0**, **-1**, **-2**, **-3**, **-4**, or **-H** (together with one of **-0**, **-1**, **-2**, **-3**, **-4**)

-g <chainName> Or one of **-c**, **-d**, **-n**, **-o**, **-t**

-q <qualifierList>

-z <databaseList>

Table 23.17.2-a:

-f <flavor>	Or one of **-0**, **-1**, **-2**, **-3**, **-4**, or **-H** (together with one of **-0**, **-1**, **-2**, **-3**, **-4**)
-g <chainName>	Or one of **-c**, **-d**, **-n**, **-o**, **-t**

1. We recommend that you use **ups modify** to edit the database files, in which case you don't need to run this command. **ups modify** also runs **ups verify** to prevent entry of database errors.

Table 23.17.2-a:

-q <qual-ifier-List>	
-z <data-baseList>	

23.17.3 All Valid Options

-? (**"-?"** for **csh**) Prints command description and option usage information to screen

-c Finds product instance chained to “current”

-d Finds product instance chained to “development”

-f <flavor> Described below under “The flavor options”.

-g <chainName> Finds product instance chained to **<chainName>**

-H <flavor> Described below under “The flavor options”.

-n Finds product instance chained to “new”

-o Finds product instance chained to “old”

-q <qualifierList> Finds product instance with the specified qualifiers (required and/or optional)

-t Finds product instance chained to “test”

-v(**vvv**) Prints out extra debugging information.

-z <databases> Specifies the database(s) to use

-Z Times the command (does not include time for sourcing of temp file for **setup/unsetup**)

Table 23.17.3-a:

-? ("-?" for csh)	Prints command description and option usage information to screen
-c	Finds product instance chained to “current”
-d	Finds product instance chained to “development”
-f <flavor>	Described below under “The flavor options”.
-g <chain-Name>	Finds product instance chained to **<chainName>**

Table 23.17.3-a:

-H <flavor>	Described below under “The flavor options”.
-n	Finds product instance chained to “new”
-o	Finds product instance chained to “old”
-q <qualifier-List>	Finds product instance with the specified qualifiers (required and/or optional)
-t	Finds product instance chained to “test”
-v(vvv)	Prints out extra debugging information.
-z <databases>	Specifies the database(s) to use
-Z	Times the command (does not include time for sourcing of temp file for **setup/unsetup**)

The flavor options

Flavor may be specified using **-f**, using **-H** in combination with any of **-0**, **-1**, **-2**, **-3**, **-4**, or just using one of **-0**, **-1**, **-2**, **-3**, **-4**. These options are not valid with each other (except **-H** with a number option).

-f <flavor> Finds product instance of specified flavor. If specified and no exact match is found, the command fails. Multiple values accepted, but **UPS** looks only at first in list.

-H <flavor> Must be used with any of **-0**, **-1**, **-2**, **-3**, **-4**. Specifies flavor and builds a flavor list for that family starting at the level specified. **UPS** finds the product instance of specified level of that flavor; e.g., **-2H IRIX+6.2** is equivalent to **-f IRIX+6**.

-0 Specifies flavor as NULL; equivalent to **-f NULL**

-1 Specifies flavor as OS value up to the generic OS; e.g., on a SunOS machine it is equivalent to **-f SunOS**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX.

-2 Specifies flavor for product instance on local and distribution nodes as OS value up to the version; e.g., equivalent to **-f SunOS+5**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.

-3 Specifies flavor for product instance on local and distribution nodes upto the release of the version; e.g., equivalent to **-f SunOS+5.6**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.2.

-4 Specifies flavor for product instance on local and distribution nodes upto the patch number of the release; e.g., equivalent to **-f SunOS+5.6.2**; if given together with **-H IRIX+6.2.1**, flavor is then specified as IRIX+6.2.1.

Table 23.17.3-b:

-f <flavor>	Finds product instance of specified flavor. If specified and no exact match is found, the command fails. Multiple values accepted, but **UPS** looks only at first in list.
-H <flavor>	Must be used with any of **-0**, **-1**, **-2**, **-3**, **-4**. Specifies flavor and builds a flavor list for that family starting at the level specified. **UPS** finds the product instance of specified level of that flavor; e.g., **-2H IRIX+6.2** is equivalent to **-f IRIX+6**.
-0	Specifies flavor as NULL; equivalent to **-f NULL**
-1	Specifies flavor as OS value up to the generic OS; e.g., on a SunOS machine it is equivalent to **-f SunOS**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX.
-2	Specifies flavor for product instance on local and distribution nodes as OS value up to the version; e.g., equivalent to **-f SunOS+5**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.
-3	Specifies flavor for product instance on local and distribution nodes upto the release of the version; e.g., equivalent to **-f SunOS+5.6**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.2.
-4	Specifies flavor for product instance on local and distribution nodes upto the patch number of the release; e.g., equivalent to **-f SunOS+5.6.2**; if given together with **-H IRIX+6.2.1**, flavor is then specified as IRIX+6.2.1.

23.17.4 ups touch Example

To illustrate this command, we first run a **ups list** showing the modified date/time and the modifier(s) for a product:

```
% ups list teledata -aKproduct:modified:modifier

   "teledata" "1999-09-08 21.44.04 GMT:1999-09-08 21.38.23 GMT"
   "olduser:olduser"
```

Now run **ups touch** to change these values:

```
% ups touch teledata
```

And verify that they have changed, by rerunning the **ups list** command:

```
% ups list teledata -aKproduct:modified:modifier

   "teledata" "1999-11-19 18.43.00 GMT:1999-09-08 21.38.23 GMT"
   newuser:olduser"
```

23.18 ups unconfigure

For any product instance whose table file includes an UNCONFIGURE action, the **ups unconfigure** command executes this action. An UNCONFIGURE action usually includes functions that reverse, approximately or fully, the functions run by the CONFIGURE action. The **ups unconfigure** command gets run by default by **ups undeclare** when the product is removed from a database (see section 23.19 *ups undeclare*, in particular the **-C** option), but can be run manually as needed.

23.18.1 Command Syntax

```
% ups unconfigure [<options>] <product> [<version>]
```

23.18.2 Commonly Used Options

See section 23.18.3 *All Valid Options* for descriptions of each option.

-f <flavor> Or one of **-0**, **-1**, **-2**, **-3**, **-4**, or **-H** (alone or together with one of **-0**, **-1**, **-2**, **-3**, **-4**)

-g <chainName> Or one of **-c**, **-d**, **-n**, **-o**, **-t**

-q <qualifierList>

-z <databaseList>

Table 23.18.2-a:

-f <flavor>	Or one of **-0**, **-1**, **-2**, **-3**, **-4**, or **-H** (alone or together with one of **-0**, **-1**, **-2**, **-3**, **-4**)
-g <chainName>	Or one of **-c**, **-d**, **-n**, **-o**, **-t**
-q <qualifierList>	

Table 23.18.2-a:

-z <data-baseList>	

23.18.3 All Valid Options

-? (**"-?"** for **csh**) Prints command description and option usage information to screen

-c Finds product instance chained to “current”

-d Finds product instance chained to “development”

-f <flavor> Described below under “The flavor options”.

-g <chainName> Finds product instance chained to **<chainName>**

-H <flavor> Described below under “The flavor options”.

-m <tableFileName> Specifies table file name

-M <tableFileDir> Specifies table file directory

-n Finds product instance chained to “new”

-o Finds product instance chained to “old”

-O "<flags>" Sets the value of $UPS_OPTIONS to **<flags>**.

-P Requires **UPS** to rely only on information supplied on the command line to locate the product instance (prevents **UPS** from searching in a database)

-q <qualifierList> Finds product instance with the specified qualifiers (required and/or optional)

-r <prodRootDir> Specifies the product root directory

-s Lists what command would do; but does not execute the command

-t Finds product instance chained to “test”

-U <upsDir> Specifies location of `ups` directory; default value is **ups**

-v(**vvv**) Prints out extra debugging information.

-V Does not delete the temporary script files or partially installed products when command finishes; instead lists them on the screen

-z <databaseList> Specifies the database(s) in which to look for the product and its dependencies

-Z Times the command (does not include time for sourcing of temp file for **setup/unsetup**)

Table 23.18.3-a:

-? ("-?" for csh)	Prints command description and option usage information to screen
-c	Finds product instance chained to “current”
-d	Finds product instance chained to “development”
-f <flavor>	Described below under “The flavor options”.
-g <chainName>	Finds product instance chained to **<chainName>**
-H <flavor>	Described below under “The flavor options”.
-m <tableFileName>	Specifies table file name
-M <tableFileDir>	Specifies table file directory
-n	Finds product instance chained to “new”
-o	Finds product instance chained to “old”
-O "<flags>"	Sets the value of $UPS_OPTIONS to **<flags>**.
-P	Requires **UPS** to rely only on information supplied on the command line to locate the product instance (prevents **UPS** from searching in a database)
-q <qualifierList>	Finds product instance with the specified qualifiers (required and/or optional)
-r <prodRootDir>	Specifies the product root directory
-s	Lists what command would do; but does not execute the command
-t	Finds product instance chained to “test”
-U <upsDir>	Specifies location of **ups** directory; default value is **ups**
-v(vvv)	Prints out extra debugging information.
-V	Does not delete the temporary script files or partially installed products when command finishes; instead lists them on the screen

Table 23.18.3-a:

-z <dataBaseList>	Specifies the database(s) in which to look for the product and its dependencies
-Z	Times the command (does not include time for sourcing of temp file for **setup/unsetup**)

The flavor options

Flavor may be specified using **-f**, using **-H** by itself or in combination with any of **-0**, **-1**, **-2**, **-3**, **-4**, or just using one of **-0**, **-1**, **-2**, **-3**, **-4**. These options are not valid with each other (except **-H** with a number option).

-f <flavor> Finds product instance of specified flavor. If specified and no exact match is found, the command fails. Multiple values accepted, but **UPS** looks only at first in list.

-H <flavor> Specifies flavor and builds a flavor list for that family starting at the level specified. Multiple values accepted, but **UPS** looks only at first in list.

Can be used alone (without an accompanying number option). In this case, **UPS** finds the best match instance for the specified flavor family.

If used with any of **-0**, **-1**, **-2**, **-3**, **-4**, **UPS** finds the product instance of specified level of that flavor; e.g., **-2H IRIX+6.2** is equivalent to **-f IRIX+6**.

-0 Specifies flavor as NULL; equivalent to **-f NULL**

-1 Specifies flavor as OS value up to the generic OS; e.g., on a SunOS machine it is equivalent to **-f SunOS**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX.

-2 Specifies flavor for product instance on local and distribution nodes as OS value up to the version; e.g., equivalent to **-f SunOS+5**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.

-3 Specifies flavor for product instance on local and distribution nodes upto the release of the version; e.g., equivalent to **-f SunOS+5.6**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.2.

-4	Specifies flavor for product instance on local and distribution nodes upto the patch number of the release; e.g., equivalent to **-f SunOS+5.6.2**; if given together with **-H IRIX+6.2.1**, flavor is then specified as IRIX+6.2.1.

Table 23.18.3-b:

-f <flavor>	Finds product instance of specified flavor. If specified and no exact match is found, the command fails. Multiple values accepted, but **UPS** looks only at first in list.
-H <flavor>	Specifies flavor and builds a flavor list for that family starting at the level specified. Multiple values accepted, but **UPS** looks only at first in list. Can be used alone (without an accompanying number option). In this case, **UPS** finds the best match instance for the specified flavor family. If used with any of **-0**, **-1**, **-2**, **-3**, **-4**, **UPS** finds the product instance of specified level of that flavor; e.g., **-2H IRIX+6.2** is equivalent to **-f IRIX+6**.
-0	Specifies flavor as NULL; equivalent to **-f NULL**
-1	Specifies flavor as OS value up to the generic OS; e.g., on a SunOS machine it is equivalent to **-f SunOS**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX.
-2	Specifies flavor for product instance on local and distribution nodes as OS value up to the version; e.g., equivalent to **-f SunOS+5**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.
-3	Specifies flavor for product instance on local and distribution nodes upto the release number of the version; e.g., equivalent to **-f SunOS+5.6**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.2.
-4	Specifies flavor for product instance on local and distribution nodes upto the patch number of the release; e.g., equivalent to **-f SunOS+5.6**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.2.

23.18.4 More Detailed Description

A product's configuration may involve creating links to the product root directory from other areas (see section 16.1.3 *Third-Party Products Requiring a Hard-Coded Path*). If the area is not identical for each node accessing the **UPS** database in which the product instance has been declared (i.e., same path but separate areas), then the **ups configure** command needs to be run manually on each node that mounts a unique area. Similarly, when removing such a product from a database, you will need to run the **ups unconfigure** command manually on each node. If you are not sure

whether you need to unconfigure a product instance on each node, look through the ACTION=UNCONFIGURE steps in the table file to see what they do.

Internal Processes

- Check node authorization
- Process the UNCONFIGURE action
- Execute the temp file

23.18.5 ups unconfigure Example

perl is a product that requires **`ups configure`** and **`ups unconfigure`** to be run manually for each machine flavor in a cluster. The sample command, which should be issued from a machine of flavor SunOS+5, runs the UNCONFIGURE action in the table file associated with the product **perl**, version v5_005 for flavor SunOS+5. If that action is not present, it undoes all the reversible functions included under the CONFIGURE action, by default.

```
% ups unconfigure perl v5_005 -f SunOS+5
```

This command should take care of the "unconfiguration" of **perl** on all the machines of flavor SunOS+5 in the cluster. A command like this, but with the appropriate flavor, must be run for each machine flavor represented in the cluster.

23.19 ups undeclare

The **`ups undeclare`** command is used for two separate purposes:

1) to remove an instance from a database (and optionally remove its product root directory); the information that gets removed includes:
   - the version file, or the portion of the version file, that pertains to the instance
   - any chain files, or the portions of any chain files, that pertain to the instance
2) to remove a chain from an instance

23.19.1 Command Syntax

For removing an instance

```
% ups undeclare <flavor_option> [<other_options>] <product> \
  <version>
```

For removing a chain

```
% ups undeclare <chain_option> [<other_options>] <product>
```

23.19.2 Commonly Used Options

See section 23.18.3 *All Valid Options* for descriptions of each option.

For removing an instance

`-f <flavor>` Or one of **`-0`**, **`-1`**, **`-2`**, **`-3`**, **`-4`**, or **`-H`** (together with one of **`-0`**, **`-1`**, **`-2`**, **`-3`**, **`-4`**)

`-g <chainName>` Or one of **`-c`**, **`-d`**, **`-n`**, **`-o`**, **`-t`**

`-q <qualifierList>`

`-y`

`-Y`

`-z <databaseList>`

Table 23.19.2-a:

`-f <flavor>`	Or one of **`-0`**, **`-1`**, **`-2`**, **`-3`**, **`-4`**, or **`-H`** (together with one of **`-0`**, **`-1`**, **`-2`**, **`-3`**, **`-4`**)
`-g <chainName>`	Or one of **`-c`**, **`-d`**, **`-n`**, **`-o`**, **`-t`**
`-q <qualifierList>`	
`-y`	
`-Y`	
`-z <databaseList>`	

For removing a chain

`-f <flavor>` Or one of **`-0`**, **`-1`**, **`-2`**, **`-3`**, **`-4`**, or **`-H`** (together with one of **`-0`**, **`-1`**, **`-2`**, **`-3`**, **`-4`**)

`-g <chainName>` Or one of **`-c`**, **`-d`**, **`-n`**, **`-o`**, **`-t`**

`-q <qualifierList>`

`-z <databaseList>`

Table 23.19.2-b:

`-f <flavor>`	Or one of **`-0`**, **`-1`**, **`-2`**, **`-3`**, **`-4`**, or **`-H`** (together with one of **`-0`**, **`-1`**, **`-2`**, **`-3`**, **`-4`**)
`-g <chainName>`	Or one of **`-c`**, **`-d`**, **`-n`**, **`-o`**, **`-t`**
`-q <qualifierList>`	
`-z <databaseList>`	

23.19.3 All Valid Options

Valid only for removing an instance (not for removing a chain)

`-y` Deletes product root directory, provides confirmation prompt

`-Y` Deletes product root directory, does not provide confirmation prompt

Table 23.19.3-a:

`-y`	Deletes product root directory, provides confirmation prompt
`-Y`	Deletes product root directory, does not provide confirmation prompt

Valid for both functions

`-?` (**`"-?"`** for **csh**) Prints command description and option usage information to screen

`-c` Finds product instance chained to “current”

-C	When removing a product: Prevents execution of the UNCONFIGURE action When removing a chain: Prevents execution of the corresponding "unchain" action
-d	Finds product instance chained to "development"
-f <flavor>	Described below under "The flavor options".
-g <chainName>	Finds product instance chained to **<chainName>**
-H <flavor>	Described below under "The flavor options".
-m <tableFileName>	Specifies table file name
-M <tableFileDir>	Specifies table file directory
-n	Finds product instance chained to "new"
-o	Finds product instance chained to "old"
-O "<flags>"	Sets the value of $UPS_OPTIONS to **<flags>**.
-q <qualifierList>	Finds product instance with the specified qualifiers (required and/or optional)
-r <prodRootDir>	Specifies the product root directory
-t	Finds product instance chained to "test"
-U <upsDir>	Specifies location of `ups` directory; default value is **ups**
-v(**vvv**)	Prints out extra debugging information.
-V	Does not delete the temporary script files or partially installed products when command finishes; instead lists them on the screen
-z <databases>	Specifies the database(s) to use
-Z	Times the command

Table 23.19.3-b:

-? ("-?" for csh)	Prints command description and option usage information to screen
-c	Finds product instance chained to "current"
-C	When removing a product: Prevents execution of the UNCONFIGURE action When removing a chain: Prevents execution of the corresponding "unchain" action
-d	Finds product instance chained to "development"
-f <flavor>	Described below under "The flavor options".

Table 23.19.3-b:

-g <chain-Name>	Finds product instance chained to **<chainName>**
-H <fla-vor>	Described below under “The flavor options”.
-m <table-FileName>	Specifies table file name
-M <table-FileDir>	Specifies table file directory
-n	Finds product instance chained to “new”
-o	Finds product instance chained to “old”
-O "<flags>"	Sets the value of $UPS_OPTIONS to **<flags>**.
-q <qual-ifier-List>	Finds product instance with the specified qualifiers (required and/or optional)
-r <pro-dRootDir>	Specifies the product root directory
-t	Finds product instance chained to “test”
-U <ups-Dir>	Specifies location of **ups** directory; default value is **ups**
-v(vvv)	Prints out extra debugging information.
-V	Does not delete the temporary script files or partially installed products when command finishes; instead lists them on the screen
-z <data-bases>	Specifies the database(s) to use
-Z	Times the command

The flavor options

Flavor may be specified using **-f**, using **-H** in combination with any of **-0**, **-1**, **-2**, **-3**, **-4**, or just using one of **-0**, **-1**, **-2**, **-3**, **-4**. These options are not valid with each other (except **-H** with a number option).

-f <flavor> Finds product instance of specified flavor. If specified and no exact match is found, the command fails.

-H <flavor>	Must be used with any of **-0**, **-1**, **-2**, **-3**, **-4**. Specifies flavor and builds a flavor list for that family starting at the level specified. **UPS** finds the product instance of specified level of that flavor; e.g., **-2H IRIX+6.2** is equivalent to **-f IRIX+6**.
-0	Specifies flavor as NULL; equivalent to **-f NULL**
-1	Specifies flavor as OS value up to the generic OS; e.g., on a SunOS machine it is equivalent to **-f SunOS**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX.
-2	Specifies flavor for product instance on local and distribution nodes as OS value up to the version; e.g., equivalent to **-f SunOS+5**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.
-3	Specifies flavor for product instance on local and distribution nodes upto the release of the version; e.g., equivalent to **-f SunOS+5.6**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.2.
-4	Specifies flavor for product instance on local and distribution nodes upto the patch of the release; e.g., equivalent to **-f SunOS+5.6.2**; if given together with **-H IRIX+6.2.1**, flavor is then specified as IRIX+6.2.1.

Table 23.19.3-c:

-f <flavor>	Finds product instance of specified flavor. If specified and no exact match is found, the command fails.
-H <flavor>	Must be used with any of **-0**, **-1**, **-2**, **-3**, **-4**. Specifies flavor and builds a flavor list for that family starting at the level specified. **UPS** finds the product instance of specified level of that flavor; e.g., **-2H IRIX+6.2** is equivalent to **-f IRIX+6**.
-0	Specifies flavor as NULL; equivalent to **-f NULL**
-1	Specifies flavor as OS value up to the generic OS; e.g., on a SunOS machine it is equivalent to **-f SunOS**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX.
-2	Specifies flavor for product instance on local and distribution nodes as OS value up to the version; e.g., equivalent to **-f SunOS+5**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.
-3	Specifies flavor for product instance on local and distribution nodes upto the release of the version; e.g., equivalent to **-f SunOS+5.6.2**; if given together with **-H IRIX+6.2.1**, flavor is then specified as IRIX+6.2.1.

Table 23.19.3-c:

`-4`	Specifies flavor for product instance on local and distribution nodes as most significant OS specification or the full specification; e.g., equivalent to **`-f SunOS+5.6`**; if given together with **`-H IRIX+6.2`**, flavor is then specified as IRIX+6.2.

23.19.4 More Detailed Description

Removing a Product Instance

Using `ups undeclare` is the recommended procedure for removing product instances. Removing them manually does not ensure that all the files get deleted or that chains get updated properly, which can lead to a fragmented products area.

To undeclare a product instance, you must specify the *version* of the instance, not its *chain*, in the **`ups undeclare`** command. **Specifying the chain removes only that chain, not the instance itself.** When an instance gets "undeclared", all information pertaining to it is removed from the **UPS** database in question; this includes:

- the version file, or the portion of the version file, that pertains to the instance
- any chain files, or the portions of any chain files, that pertain to the instance

You can also opt to remove the product instance's directory tree starting from its root directory. To do so, use one of the options **`-y`** or **`-Y`** (**`-y`** queries you for confirmation, **`-Y`** does not).

Before removing anything, you should find out if any other products have the product instance in question declared as a dependency.[1] If so, you may want to reconsider removing it. Removal of the product instance may affect the operation of its parent products.

We recommend always including a flavor option if you have a multi-flavor database.

The **`ups undeclare`** command executes **`ups unconfigure`** by default (the UNCONFIGURE process can be suppressed by using the **`-C`** option, however normally you want this process to execute).

1. The **`ups parent`** command will provide this information. The command is not available in **UPS** version v4; it is planned for a future release.

Special case: If a product has a CONFIGURE action that modifies files outside of its product root directory, and if this instance is used by more than one node, flavor or file system, then you may need to run **`ups undeclare`** or **`ups unconfigure`** on all of the nodes before removing the product files on any node. Check the product's table file.

Removing a Chain from an Instance

To remove a chain, include the chain specification in the **`ups undeclare`** command, but do not include the version. Including both the chain and version is bound to be either redundant or incompatible, and may result in removing the product declaration! We recommend always including the **`-f <flavor>`** option if you have a multi-flavored database.

Internal Processes

- Find database to use
- If necessary process all appropriate 'UNCHAIN' actions
- Process the UNCONFIGURE action
- Process the UNDECLARE action
- If necessary delete the product's home area
- Execute the temp file

23.19.5 ups undeclare Examples

Undeclare an instance

```
% ups undeclare -f IRIX+5 tcl v7_6a -y
```

In this example, we undeclare and remove current instance (by default) of the product **tcl** v7_6a for the flavor IRIX+5 (**`-f`** option) from the default database. Notice that the version is included for instance identification as required. We opt to remove the product root directory after query (lowercase **`-y`** option):

```
Product home directory -
        /export/home/t1/aheavey/upsII/products/tcl/v7_6a/
Delete this directory?y
```

We respond "y" for yes. To respond no, we would enter "n".

Undeclare a chain

```
% ups undeclare -c ximagetools -f NULL
```

In this command, we remove the "current" chain (**-c** option) from the instance of **ximagetools** declared as current for the flavor NULL (**-f** option). Notice that the version is not included!

If multiple flavor/qualifier pairs share the chain file in question (in which case you need to specify the flavor/qualifier information on the command line), only the portion of the chain file pertaining to the specified instance will get removed; the file itself will not be deleted.

23.20 ups verify

The **ups verify** command checks the integrity of the database files for the specified product(s), and lists any errors and inconsistencies that it finds.

The **ups verify** command gets run by **ups modify** before and after file editing (see section 23.12 *ups modify*). **ups verify** can also be run manually as needed.

23.20.1 Command Syntax

```
% ups verify [<options>] [<product>] [<version>]
```

23.20.2 Commonly Used Options

See section 23.20.3 *All Valid Options* for descriptions of each option.

-f <flavorList> Or one of **-0**, **-1**, **-2**, **-3**, **-4**, or **-H** (alone or together with one of **-0**, **-1**, **-2**, **-3**, **-4**)

-g <chainName> Or one of **-c**, **-d**, **-n**, **-o**, **-t**

-q <qualifierList>

-z <databaseList>

Table 23.20.2-a:

-f <flavorList>	Or one of **-0**, **-1**, **-2**, **-3**, **-4**, or **-H** (alone or together with one of **-0**, **-1**, **-2**, **-3**, **-4**)
-g <chainName>	Or one of **-c**, **-d**, **-n**, **-o**, **-t**
-q <qualifierList>	

Table 23.20.2-a:

-z <data-baseList>	

23.20.3 All Valid Options

-? (**"-?"** for **csh**) Prints command description and option usage information to screen

-a Verifies files for all instances that match the other options given on command line

-c Finds product instance chained to “current”

-d Finds product instance chained to “development”

-f <flavorList> Described below under “The flavor options”.

-g <chainName> Finds product instance chained to **<chainName>**

-H <flavorList> Described below under “The flavor options”.

-m <tableFileName> Specifies table file name

-M <tableFileDir> Specifies table file directory

-n Finds product instance chained to “new”

-o Finds product instance chained to “old”

-q <qualifierList> Finds product instance with the specified qualifiers (required and/or optional)

-r <prodRootDir> Specifies the product root directory

-t Finds product instance chained to “test”

-U <upsDir> Specifies location of `ups` directory; default value is **ups**

-v(**vvv**) Prints out extra debugging information.

-V Does not delete the temporary script files or partially installed products when command finishes; instead lists them on the screen

-z <databaseList> Specifies the database(s) in which to look for the product and its dependencies

-Z Times the command (does not include time for sourcing of temp file for **setup/unsetup**)

Table 23.20.3-a:

-? ("-?" for csh)	Prints command description and option usage information to screen
-a	Verifies files for all instances that match the other options given on command line
-c	Finds product instance chained to “current”
-d	Finds product instance chained to “development”
-f <flavorList>	Described below under “The flavor options”.
-g <chainName>	Finds product instance chained to **<chainName>**
-H <flavorList>	Described below under “The flavor options”.
-m <tableFileName>	Specifies table file name
-M <tableFileDir>	Specifies table file directory
-n	Finds product instance chained to “new”
-o	Finds product instance chained to “old”
-q <qualifierList>	Finds product instance with the specified qualifiers (required and/or optional)
-r <prodRootDir>	Specifies the product root directory
-t	Finds product instance chained to “test”
-U <upsDir>	Specifies location of `ups` directory; default value is **ups**
-v(vvv)	Prints out extra debugging information.
-V	Does not delete the temporary script files or partially installed products when command finishes; instead lists them on the screen
-z <databaseList>	Specifies the database(s) in which to look for the product and its dependencies
-Z	Times the command (does not include time for sourcing of temp file for **setup/unsetup**)

The flavor options

Flavor may be specified using **-f**, using **-H** by itself or in combination with any of **-0**, **-1**, **-2**, **-3**, **-4**, or just using one of **-0**, **-1**, **-2**, **-3**, **-4**. These options are not valid with each other (except **-H** with a number option).

-f <flavorList> Finds product instance of specified flavor(s). If specified and no exact match is found, the command fails. Multiple values accepted.

-H <flavorList> Specifies flavor and builds a flavor list for that family starting at the level specified. Multiple values accepted.

Can be used alone (without an accompanying number option). In this case, **UPS** finds the best match instance for the specified flavor family.

If used with any of **-0**, **-1**, **-2**, **-3**, **-4**, **UPS** finds the product instance of specified level of that flavor; e.g., **-2H IRIX+6.2** is equivalent to **-f IRIX+6**.

-0 Specifies flavor as NULL; equivalent to **-f NULL**

-1 Specifies flavor as OS value up to the generic OS; e.g., on a SunOS machine it is equivalent to **-f SunOS**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX.

-2 Specifies flavor for product instance on local and distribution nodes as OS value up to the version; e.g., equivalent to **-f SunOS+5**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.

-3 Specifies flavor for product instance on local and distribution nodes upto the release of the version; e.g., equivalent to **-f SunOS+5.6**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.2.

-4 Specifies flavor for product instance on local and distribution nodes upto the patch number of the release; e.g., equivalent to **-f SunOS+5.6.2**; if given together with **-H IRIX+6.2.1**, flavor is then specified as IRIX+6.2.1.

Table 23.20.3-b:

-f <flavorList>	Finds product instance of specified flavor(s). If specified and no exact match is found, the command fails. Multiple values accepted.

Table 23.20.3-b:

-H <flavorList>	Specifies flavor and builds a flavor list for that family starting at the level specified. Multiple values accepted. Can be used alone (without an accompanying number option). In this case, **UPS** finds the best match instance for the specified flavor family. If used with any of **-0**, **-1**, **-2**, **-3**, **-4**, **UPS** finds the product instance of specified level of that flavor; e.g., **-2H IRIX+6.2** is equivalent to **-f IRIX+6**.
-0	Specifies flavor as NULL; equivalent to **-f NULL**
-1	Specifies flavor as OS value up to the generic OS; e.g., on a SunOS machine it is equivalent to **-f SunOS**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX.
-2	Specifies flavor for product instance on local and distribution nodes as OS value up to the version; e.g., equivalent to **-f SunOS+5**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.
-3	Specifies flavor for product instance on local and distribution nodes upto the release of the version; e.g., equivalent to **-f SunOS+5.6**; if given together with **-H IRIX+6.2**, flavor is then specified as IRIX+6.2.
-4	Specifies flavor for product instance on local and distribution nodes upto the patch of the release; e.g., equivalent to **-f SunOS+5.6.2**; if given together with **-H IRIX+6.2.1**, flavor is then specified as IRIX+6.2.1.

23.20.4 ups verify Example

```
% ups verify -z $MYDB teledata
```

For this example, we have given an erroneous value to the TABLE_FILE keyword in the version file for this product. The command output shows:

```
ERROR: No instance matches were made between the
version                                                 file
(/home/t1/aheavey/upsII/declared/teledata/v1_0.version)  and
the
table file (v1_1.table) for flavor (NULL) and qualifiers ()
ERROR:            Possible            UPS            database
(/home/t1/aheavey/upsII/declared) corruption in pro
duct 'teledata'.
ERROR: No instance matches were made between the chain file
(/home/t1/aheavey/up
sII/declared/teledata/current.chain)  and  the  version  file
(v1_0.version)
ERROR:            Possible            UPS            database
(/home/t1/aheavey/upsII/declared) corruption in pro
duct 'teledata'.
```